Vital Information 1

A Little-known Secret to Success

This book will help you to begin productive work with Pascal in a matter of minutes; but, like any valuable tool, you must know how to use it correctly. If you will simply invest a few *minutes* reading the next few pages, you may save *hours* of future reading time.

Are You in a Hurry?

Yes! If so, read this chapter and then turn directly to the Task Reference. There you will find all of the essential instructions for using the Pascal Workstation. Follow the directions at the beginning of this section and you'll be up and running in no time, but....

Keep in mind that the Task Reference is just a set of stripped-down, bare-bones procedures: no examples or explanations are provided. That section is designed as a rapid reference for people who are already reasonably familiar with computer programming. It tells you *how* to do something, assuming you already know *what* it is you want to do.

If you are unsure of terminology or confused by a particular operation, just look at the page you're on. The heading called "Additional Information" will tell you where to go for a gentler introduction to the material.

Now finish reading this chapter, and then turn to the Task Reference.

Not Really If you're *not* in a hurry, work through the information in this book in the order it is presented. Set the book down beside your computer, get out the discs that contain your Pascal system, work through the book and enjoy yourself. Because this book is a "hands on" companion to your computer, you needn't worry about reading for a long time before you actually get to do something. In just a few minutes you'll be asked to put the book down and try out the examples for yourself.

One Assumption

This book makes only one assumption: that your computer is already set up and all of your peripheral devices (e.g., disc drive, printer) are connected to it. If you haven't done this yet, follow the instructions in the Installation Card or Installation Guide that came with your computer, and come back when you're finished.

One Recommendation

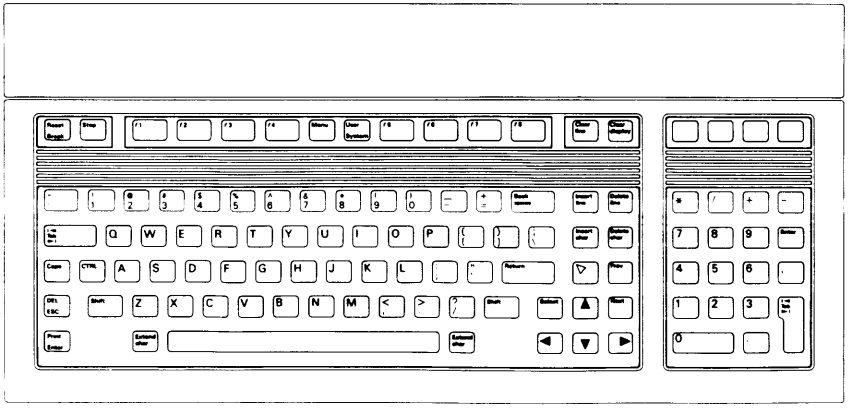
You should have at least 524 288 bytes of random access memory (RAM) installed in your computer (i.e., 512 Kbytes). Although Pascal can function with less memory, it may be less convenient to use. If you would like a complete breakdown of Pascal's memory requirements, see Appendix C.

A Word About Keys

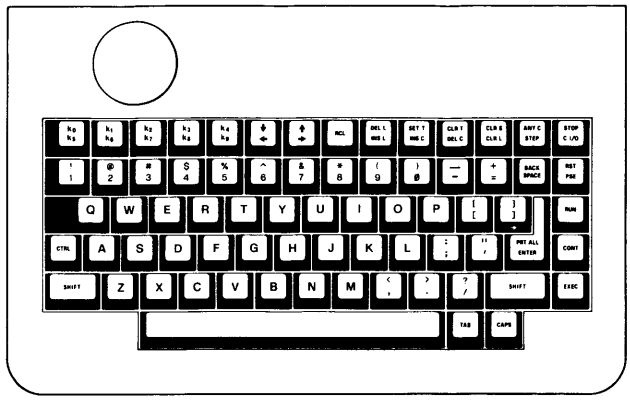
Hewlett-Packard makes several different keyboards for the Series 200/300 computers, and each uses different key labeling conventions. All three keyboards are documented in this manual, but for the sake of simplicity, the HP 46020 (or HP 46021) keyboard has been adopted as the standard in all examples and procedures. If you have one of the other keyboards, use the Key Correspondence Table at the end of this chapter to decide which key you should press.

Using the Key Correspondence Table

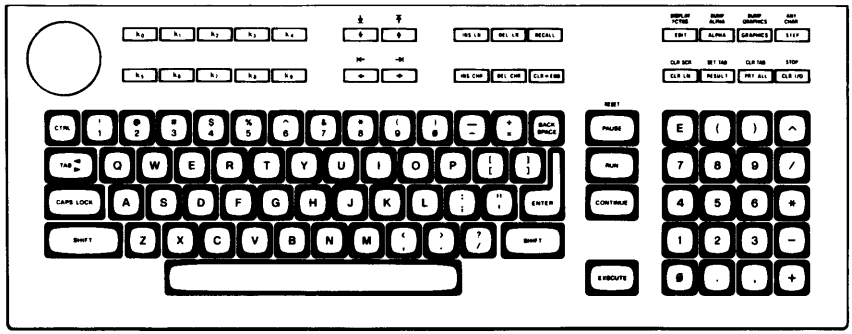
To use the Key Correspondence Table, first decide which keyboard you have. Match your keyboard to one of the illustrations below.



HP 46020 / HP 46021 Keyboards



HP 98203A Keyboard



HP 98203B/C Keyboards

If you have the HP 46020 or HP 46021 keyboard, you don't need the key correspondence table; simply press keys as instructed in this manual.

If you are using one of the other keyboards, get out the Key Correspondence Table (two copies are provided at the end of this chapter) and place it next to your computer. Find the column in the table that corresponds to your keyboard: column 2 for the HP 98203B (or HP 98203C) keyboard or column 3 for the HP 98203A keyboard.

Whenever you are told to press a key, find the key in column 1 and press instead the corresponding key in column 2 or 3. For example, suppose the book tells you to press the [Select] key. The actual key you should press is:

- [Select] if using the HP 46020 or 46021 keyboard
- [EXECUTE] if using the large keyboard (column 2)
- [EXEC] if using the small keyboard (column 3)

Going Home

Finally, before you get into Pascal, it's a good idea to know how to get out of it. All procedures and demonstrations in this book assume that you start from "home base" in the Pascal Workstation. You will know when you are at home base, also known as the **Main Command Level**, when the following prompt is shown across the top of the screen:

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

As you begin to experiment with Pascal, you may find yourself trapped in some unchartered territory, wondering how to get out. There is a way back, but you must proceed cautiously.

Pressing the [Stop] key (almost always) immediately aborts whatever you're doing and returns you to the Main Command Level. While this may be what you wanted, it can wreak havoc if:

- you're in the middle of saving something on your disc. For this reason, **never press [Stop] while your disc drive's "in use" light is on.** You may scramble the disc if you do. Instead, wait for the light to go off, and then press [Stop].
- you're in the middle of an editing session with [Stop]. You may lose everything you just typed in.

With these two exceptions, you may use [Stop] to return home at any time.

File Systems

The Pascal Workstation supports four different types of HP file systems.

- **WS1.0** — Workstation 1.0 — This file system is compatible with the UCSD definition. It was the first file system supported by the HP Pascal 1.0 Workstation.
- **LIF** — Logical Interchange Format — This is the commonest HP file system for the Pascal Workstation and also the BASIC Workstation. It is supported on most HP disc drives.
- **SRM** — Shared Resource Management — The SRM is a file server and printer/plotter spooling system that can be accessed by the Pascal and the BASIC Workstations.
- **HFS** — Hierarchical File System — This file system allows the BASIC Workstation, the Pascal Workstation, and the HP-UX Workstation to access the same disc.

In general, this manual assumes you are using LIF and notes differences when they occur. Special file system dependencies for SRM and HFS are described in the appendices.

Thanks for Listening

It's time to move on, either to the Task Reference or to Chapter 2. Whatever your path, we hope your introduction to Pascal is an enjoyable one. Please write if it isn't, and be specific about where you ran into trouble. Use the Reader Reply Card at the beginning of this book.

THANK YOU

Key Correspondence Table

HP 46020A/21A Keyboard Label	HP 98203B/C Keyboard Label	HP 98203A Keyboard Label	HP 46030A[1] Keyboard Label
Return	ENTER	ENTER	**Enter**
Select	EXECUTE	EXEC	**Ctrl-Enter**
Clear line	CLR LN	CLR L	**Ctrl-End**
Break	PAUSE	PSE	**F9**
ESC	SHIFT-EXECUTE	SHIFT-EXEC	**ESC**
CONT[2]	CONTINUE	CONT	**F8**
Reset	SHIFT-PAUSE	SHIFT-PSE	**Alt 1**
Stop	SHIFT-CLR I/O	SHIFT-C I/O	**F7**

[1] When used with the the HP 82311A product (Pascal Workstation PC). Otherwise, use the keys listed for the HP 46020A/21A keyboards.

[2] **CONT** is available on a softkey.

Removable Copy of the Key Correspondence Table

HP 46020A/21A Keyboard Label	HP 98203B/C Keyboard Label	HP 98203A Keyboard Label	HP 46030A[1] Keyboard Label
Return	ENTER	ENTER	**Enter**
Select	EXECUTE	EXEC	**Ctrl-Enter**
Clear line	CLR LN	CLR L	**Ctrl-End**
Break	PAUSE	PSE	**F9**
ESC	SHIFT-EXECUTE	SHIFT-EXEC	**ESC**
CONT[2]	CONTINUE	CONT	**F8**
Reset	SHIFT-PAUSE	SHIFT-PSE	**Alt 1**
Stop	SHIFT-CLR I/O	SHIFT-C I/O	**F7**

[1] When used with the the HP 82311A product (Pascal Workstation PC). Otherwise, use the keys listed for the HP 46020A/21A keyboards.

[2] **CONT** is available on a softkey.

Loading Pascal 2

The Pascal Language System is designed like a well-structured Pascal program. It has a main program, named the Main Command Level, which "calls" various subprograms and functions that compose the system. Each subprogram, in turn, may call a number of its own subprograms. Your role in this scheme is to tell the Pascal system which subprogram you want to run next by issuing commands from the keyboard.

If the entire set of Pascal system subprograms were required to reside in your computer's memory at one time, your computer would need well over one megabyte of RAM (Random Access Memory). Fortunately, you need initially load only the *main* programs — the Main Command Level and its support — into your computer. From there, you can selectively load the portions of the system you want to use.

All of the programs that support the Main Command Level are contained on the boot disc. This chapter is concerned with the process (which is mostly automatic) of loading the contents of the boot disc into your computer.

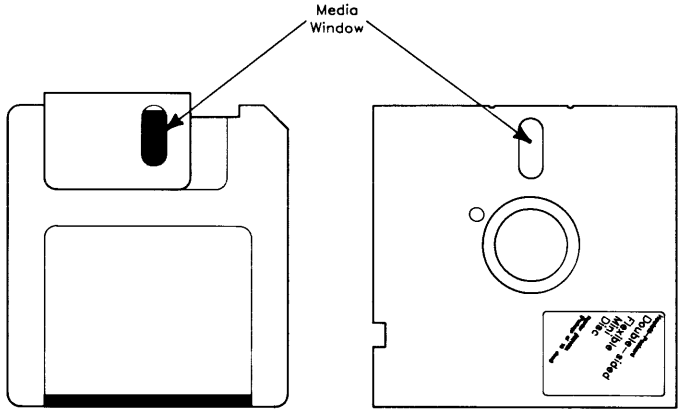
This chapter describes:

- How to properly handle flexible discs
- How to insert flexible discs into your disc drive
- How to load the Pascal Workstation from the boot disc into your computer

Flexible Disc Handling Guidelines

If this is your first experience with flexible discs, skim the following disc handling guidelines before attempting to load Pascal. Your discs represent a considerable investment, so treat them with due respect. Even a little carelessness in handling can permanently damage or dramatically reduce the life of a disc.

Before reading the guidelines, determine which type of disc you're using: either the 3½-inch *micro* flexible discs, or the 5¼-inch *mini* flexible discs. Look on the disc label, and use the following illustrations to help you.



3½-inch Micro Discs **5¼-inch Mini Discs**

The guidelines are:

Back up Discs Frequently

There is always a chance of losing information stored on a disc anytime you are using your computer. There are many possible causes for this: a programming bug, operator error, power failure, hardware failure, or disc media failure from wear out, damage or contamination. **The only sure protection against information loss is to back up your discs frequently**. You'll learn how to back up a disc in Chapter 3.

Operate Your Computer in a Clean Environment

Keep your discs away from contaminants like chemical vapors, dust, smoke, ashes, eraser crumbs and food particles. Never attempt to blow small particles from the disc; the disc cleans itself as it rotates in the disc drive.

Cover the Media Window When Not in Use

This is the single most important thing to remember about handling your disc. Micro discs have a metal slide (disc guard) that covers the media window; mini discs have a protective storage envelope. Cover the window to protect the media against dust, finger prints and scratches.

Maintain Proper Temperature and Humidity

The proper operating range is 10° C (45° F) to 45° C (115° F) and 20% to 80% relative humidity. Although the disc will operate outside these normal ranges, it will wear out faster and be more prone to errors.

Avoid Magnetic Fields

Data is stored on the disc magnetically, and can be erased by an external magnetic field. Avoid placing the disc near power transformers, magnets, large disc memories or motors.

Remove the Disc from the Drive When Not in Use

Remove the disc completely from the drive when you are through using it. Store it upright in a dust-free container.

Use a Felt Tip Pen to Label Your Disc

Use a soft felt tip pen to label your disc, and be careful to write only in the label area. Using a hard tip pen, such as a ball point, can damage the media.

Don't Touch the Surface of the Disc

Be careful not to touch the media surface through the media window. The thickness of a fingerprint is enough to cause errors. The oil in a fingerprint will also collect dust and cause a disc to wear out prematurely.

Don't Bend or Fold the Disc

The disc is flexible but will not operate if creased. Using ball point pens, rubber bands, paper clips, etc. can crease the disc.

Don't Attempt to Clean the Disc

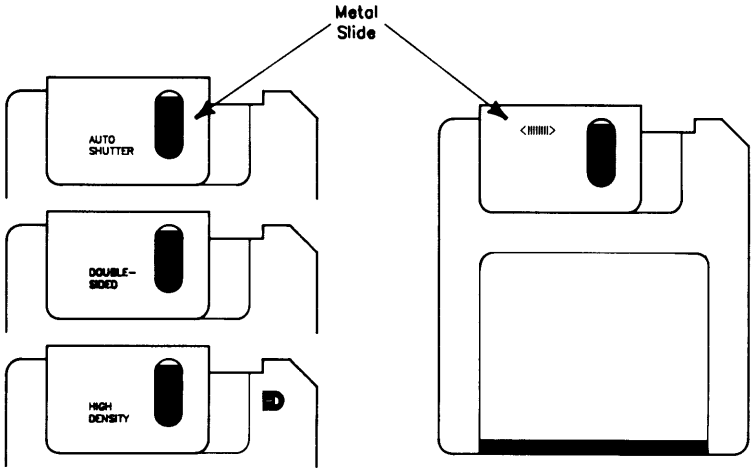
The inside surface of the disc jacket is covered with a special material that cleans the disc as it rotates in the drive. Any other method of cleaning—including blowing dust from the surface—may damage the media and cause information loss. **If a flexible disc becomes dirty or scratched, immediately transfer the data to a new disc and dispose of the old one.**

Inserting a Disc Into Your Disc Drive

Inserting a disc into a disc drive is no obvious matter if you have never done it before. Instructions are provided for both mini and micro discs; read only the section that pertains to you.

Inserting a Micro Disc

Micro discs come in two varieties: "auto shutter" and "manual shutter." To find out which type you have, look at the metal slide. If the words "AUTO SHUTTER," "DOUBLE-SIDED", "HIGH DENSITY", or "HD" appear just above the HP logo, you have an auto shutter disc. If a two-directional arrow appears, you have a manual shutter disc.



Auto Shutter Discs **Manual Shutter Disc**

CAUTION

If your disc drive ever destroys the media on a disc, stop using the drive until it can be serviced. This is exceptionally important, as continued use of the drive will destroy more media. Immediately call your nearest HP Sales and Service Office.

Note

When possible, turn the disc drive on before inserting a disc.

Inserting an Auto Shutter Disc

Slide the disc into the disc drive, label side up and metal slide facing the disc drive (try it!).

The drive should completely "swallow" the disc and may flash its light to signal acceptance. If the drive rejects the disc and pushes it back out at you, follow these steps:

1. Remove the disc and make sure you inserted it label side up. Also check that the disc drive is turned on. If you inserted the disc incorrectly, try it again.
2. If you inserted the disc correctly but the drive still refuses it, see if you can move the metal slide to the left, exposing the media. If the slide works, reinsert the disc.

Inserting a Manual Shutter Disc

To insert manual shutter micro discs into your disc drive, follow this procedure (try it!):

1. Move the metal slide all the way to the left, exposing the media.
2. Insert the disc into the drive, label-side up and metal end facing the disc drive.

The drive should completely "swallow" the disc and flash its light to signal acceptance. If it pushes the disc back out at you, check that the label is facing up and the drive is turned on. Reinsert the disc.

Removing a Micro Disc

Make sure the disc drive is turned on and press the disc eject button in the lower-right corner of the drive (try it!). Remove the disc and close the metal slide over the media window if it is not already covered. To close auto shutter discs, pinch the corner to the left of the metal slide (marked "PINCH").

Inserting a Mini Disc

Note

When possible, turn the disc drive on before inserting a disc.

To insert a mini disc into your disc drive, follow this procedure (try it!):

1. Open the drive door by lifting the door handle. Check that there is not already a disc in the drive.
2. Insert the disc into the drive, label-side up with the media window facing the drive.
3. Close the drive door. If the door will not close, push the disc farther into the drive and try again.

CAUTION

If you accidentally insert another disc when one is already in the drive, remove the bottom disc first. Otherwise, the disc drive could be damaged.

Removing a Mini Disc

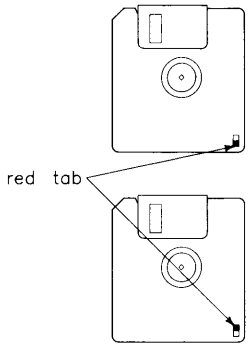
To remove a mini disc from your disc drive, follow this procedure (try it!):

1. Make sure the disc drive is turned on.
2. Lift the drive door and carefully pull the disc out.
3. Return the disc to its protective envelope and store it upright in a dust-free box.
4. Close the drive door.

Write-enabling/protecting Flexible Discs

3½-inch Double-sided Micro discs

Gray double-sided micro discs are write-enabled when you receive them. That means you can write data on the disc at any time. When you want to ensure that data on the disc is not accidentally overwritten or erased, you can write-protect the disc. The write-enabling/protecting mechanism works like this:



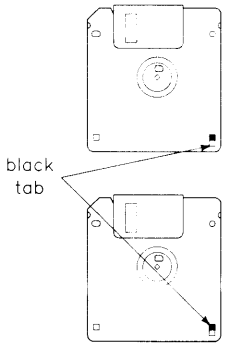
To write-protect the disc, make sure you can see through the write-protect hole in the corner of the disc—the tab should **not** cover the hole.

To write-enable the disc, make sure the tab covers the write-protect hole in the corner of the disc—you should not be able to see through the hole.

Double-sided Micro disc Write-enabling/protecting

3½-inch High-density Micro discs

Black high-density micro discs are write-enabled when you receive them. That means you can write data on the disc at any time. When you want to ensure that data on the disc is not accidentally overwritten or erased, you can write-protect the disc. The write-enabling/protecting mechanism works like this:



To write-protect the disc, make sure you can see through the write-protect hole in the corner of the disc—the tab should **not** cover the hole.

To write-enable the disc, make sure the tab covers the write-protect hole in the corner of the disc—you should not be able to see through the hole.

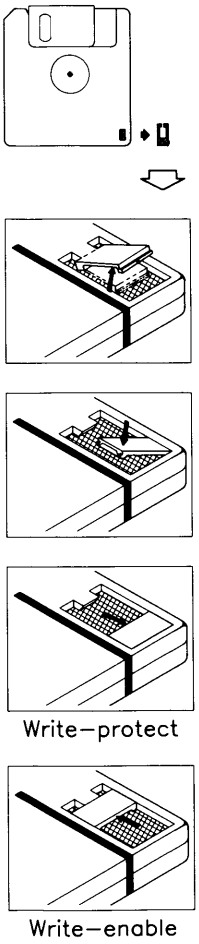
High-density Micro disc Write-enabling/protecting

The second hole on the high-density disc just opposite the write-protect hole is used for identifying the disc as a high-density disc. The high-density indicator hole should never be covered up.

3½-inch Single-sided Micro discs

Blue single-sided micro discs are write-enabled when you receive them. That means you can write data on the disc at any time. When you want to ensure that data on the disc is not accidentally overwritten or erased, you can write-protect the disc.

The write-enabling/protecting mechanism works like this:



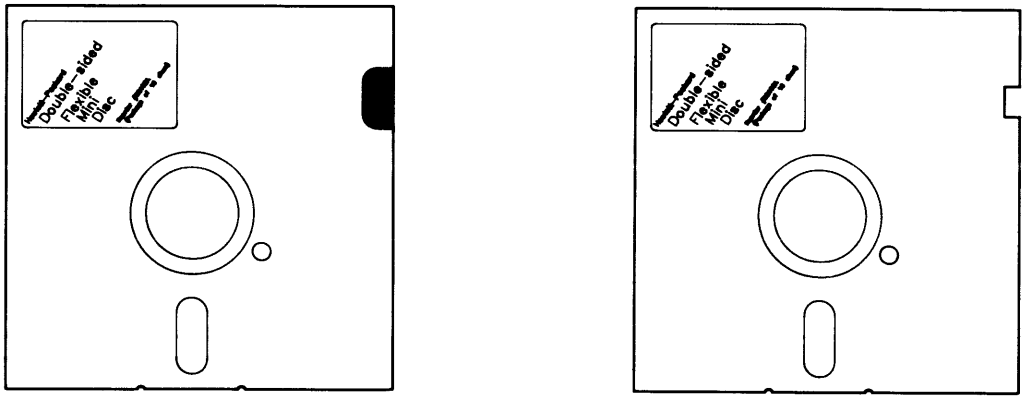
1. Weaken or score the attached point so as not to break the tab. Break off the write-protect tab.
2. Align the protrusion on the tab with the groove in the disc.
3. Depress the tab into the groove—tab should fit snugly. In the position shown, the tab write-protects the disc.
4. To write-enable the disc, slide the tab up.

3½-inch Micro Discs

5¼-inch Mini discs

Covering or uncovering a notch in the disc jacket determines whether the disc drive can write information on the disc. When the notch is covered, it's impossible for the drive to write on the disc; thus information already on the disc is protected from being overwritten or erased.

Labels are provided with discs to allow you to cover the write-protect notch.



Write-protected **Write-enabled**

5¼-inch Mini Disc Write Protection

Booting the Pascal Workstation

The process of loading a language system from a disc into your computer's memory is called **booting**.

The following procedure describes how to boot the Pascal Workstation using flexible discs. If you have any problems, refer to the section "Solving Booting Problems" following this procedure.

Note that if your Workstation is part of a Shared Resource Manager (SRM) system, your system manager may have installed the Pascal Workstation on the SRM system disc and set up an automatic bootup scheme for you. If this is the case, you boot the Pascal Workstation simply by turning your workstation off and then on. If some other system boots, see your system manager for instructions.

To boot Pascal:

1. Turn your computer off.
2. Turn **all** of your disc drives on. If you don't turn them on now, the Pascal Workstation will not recognize them.
3. Find the appropriate boot disc:
   - If you have a Series 300 computer with a bit-mapped dispaly card, find the disc labeled *BOOT2:*.
   - If you have a Series 200 computer or a Series 200 computer with an HP 98546A display card, find the disc labeled *BOOT:*.
4. Find unit #3: in the **Unit Number** column of the Pascal Workstation Worksheet. (Found at the beginning of the *Peripheral Installation Guide.*)
5. Insert the boot disc into the flexible disc drive that has been assigned unit #3:.
6. If you have dual flexible disc drives, insert the disc marked *SYSVOL:* into the flexible disc drive that has been assigned unit #4:. Make sure *SYSVOL:* is write-enabled (see previous section).
7. Turn your computer on.
8. Follow the instructions on the screen:
   - If you have dual drives, the booting process is automatic. No further action is required.

- If you have only one flexible disc drive, you will be instructed to:

```
Please put SYSVOL in unit #3
and press the X key...
```

  Remove the boot disc, insert the disc labeled *SYSVOL:*, and press the [X] key. Make sure *SYSVOL:* is write-enabled (see previous section).

9. Wait for a display similar to the one shown below. The Pascal Workstation has now been booted.

```
New system date ?

  System date is          15-Feb-87
  Clock time is            0: 8:39
  Time Zone is             7: 0: 0

  Workstation             Rev. 3.2  15-Jan-87

  Available Global Space 62098 bytes
  Total Available Memory 733688 bytes

  System  volume:  SYSVOL:
  Default volume:  BOOT2:



Copyright 1987 Hewlett-Packard Company
All rights are reserved.  Copying or other
reproduction of this program except for archival
purposes is prohibited without prior
written permission of Hewlett-Packard Company.
```

Pascal Workstation's Initial Display.

Solving Booting Problems

If Pascal failed to boot, this section will help you find and correct the problem. If the initial display appeared as expected, move on to Chapter 3.

Check the following list for an explanation of common booting errors and how to recover from them. If your situation is not listed there, call HP for help.

SEARCHING FOR A SYSTEM

If this message remains in the lower-left corner of the screen for two minutes or more, the computer cannot find the Pascal System. Possible causes for this problem are:

- The disc drive door is not closed.
- The incorrect disc was inserted into unit #3:. Make sure you use the disc labeled BOOT: or BOOT2:.
- The disc drive is not turned on.
- The disc drive containing the boot disc is not connected to the computer.
- You have two disc drives set to the same bus address on the same HP-IB interface (see your *Peripheral Installation Guide*).
- A misaligned flexible mini disc. Remove the disc and make sure the center hole is properly aligned. If its not, use a pencil to carefully shift the disc within its protective case. Do not touch the medium with the pencil. Reinsert the disc.
- You have a printer connected to the HP-IB that is set in "listen always" mode. Refer to the printer's manual and reset this switch.

Correct the problem and press the [Reset] key, or turn the computer off, then on again.

IORESULT,ERROR = 0 112

This message tells you that there was insufficient memory in your computer to load the Pascal Workstation. To recover:

1. Remember that the minimum recommended memory size for Pascal is 524 288 (512K) bytes.
2. Remove the boot disc and turn your computer on. The amount of memory present in the computer is displayed on the screen.
3. If you have memory cards installed that are not registering, turn the computer off, remove the cards, and check the address switch settings on each one. Check your *Peripheral Installation Guide* for correct switch settings. Replace the cards and turn the computer on. If all of the memory still fails to register, call HP for help.
4. If you have insufficient memory in your computer for the Pascal Workstation, contact your HP Sales Representative to purchase additional memory cards.

UNABLE TO FIND SYSTEM RESET TO RETRY

This message tells you that the computer could not find the Pascal Workstation. Possible causes of this error are:

- The disc drive door is not closed.
- The incorrect disc was inserted into unit #3:. Make sure you use the disc labeled BOOT: or BOOT2:.
- The disc drive is not turned on.
- The disc drive containing the boot disc is not connected to the computer.
- You have two disc drives set to the same bus address on the same select code.
- A misaligned flexible mini disc. Remove the disc and make sure the center hole is properly aligned. If its not, use a pencil to carefully shift the disc within its protective case. Do not touch the medium with the pencil. Reinsert the disc.

Correct the problem and press the [Reset] key, or turn the computer off, then on again.

SYSTEM NOT FOUND

This message may be displayed at the bottom-left corner of the screen. It occurs when you have multiple systems on-line, and have tried to select the Pascal Workstation for booting (see your computer's *Installation Guide* or *Installation Reference* for details of booting with multiple systems present).

Press [Reset] to restart the booting process. If the message recurs, possible sources of error are:

- The disc drive door is not closed.
- The incorrect disc was inserted into unit #3:. Make sure you use the disc labeled BOOT: or BOOT2:.
- The disc drive is not turned on.
- The disc drive containing the boot disc is not connected to the computer.
- The space bar (signalling that you want to choose which system to boot) was pressed too soon. Wait until the `Keyboard` message is displayed in the self-test list before pressing the space bar.

Correct the problem and press the [Reset] key, or turn the computer off, then on again.

FLOPPY ERROR #88,1 READ .. CRC ERROR RESET TO RETRY

This message tells you that your boot disc is damaged. Press the [Reset] key, or turn the computer off, then on again. If the message recurs, call HP for help.

Notes:

A Few Preliminaries 3

Now that you have the Pascal Workstation running on your computer, there are a few preliminary tasks to perform before you program in Pascal or run an application. These tasks include:

- Setting the date and time — not required if your workstation is connected to a Shared Resource Management (SRM) system.
- Preparing your discs for use — there is a one-time formatting (initialization) process each disc must undergo before it can be used.
- Making a "back up" copy of the Pascal software — having an archive copy of your operating system is a very good idea.

What Time is it?

Pascal 3.2 lets you specify the *time of day* and the *time zone*. When you do, your Pascal Workstation can compute *Greenwich Mean Time* (GMT) which is then used to date the files it creates. This is especially important if you plan to use HP-UX on the same workstation hardware, or share HFS discs with an HP-UX system. Later, when you set the time zone, you will need to enter the number of hours you would have to move your local clock ahead (positive) or back (negative) to reach GMT. Values shown are for standard time (subtract one for daylight savings time).

Time Zone	City	Value
	Honolulu	10
Pacific	San Francisco	8
Mountain	Denver	7
Central	Chicago	6
Eastern	New York	5
	London	0
	Paris	-1
	Cairo	-2
	Hongkong	-8
	Tokyo	-9
	Sydney	-10

Setting the Date and Time

In the last chapter, you booted the Pascal system from the BOOT: disc into your computer. You know the system was successfully booted because the following display now appears on the screen:

```
New system date ?

  System date is          15-Feb-87
  Clock time is            0: 8:39
  Time Zone is             0: 0: 0

  Workstation             Rev. 3.2  15-Jan-87

  Available Global Space 62098 bytes
  Total Available Memory 733688 bytes

  System  volume:  SYSVOL:
  Default volume:  BOOT:



Copyright 1987 Hewlett-Packard Company
All rights are reserved.  Copying or other
reproduction of this program except for archival
purposes is prohibited without prior
written permission of Hewlett-Packard Company.
```

Initial Display

Note

If this type of display does *not* appear, press [Stop],and then [V].

Entering the Date

Notice the prompt at the top of the screen:

```
New system date ?
```

The computer is asking you to type in the current date. First check if the date already entered is correct. The date is displayed directly below the prompt and looks like this:

```
System date is        1-Jan-70
```

If the date is correct, simply press [Return] to accept it. If the system date is not today's date, correct it like this:

1. Type the date in the form **dd-mmm-yy** where:
   - **dd** is one or two digits representing the day
   - **mmm** is the first three letters of the month
   - **yy** is the last two digits of the year.

   For example, if today is the **7**th of **Jan**uary, **1987**, you could type:

   ```
   7-JAN-87
   ```

   If You Make a Mistake... Simply press the [Back space] key until you erase your error. Then type the date over again.

2. Press [Return] to register the new date.

   If You Make a Mistake... Simply press [Return] twice and then the [V] key. The date prompt will reappear on the screen. Type the date over again and press [Return].

Entering the Time

After you enter the date, the following prompt appears at the top of the screen:

```
New system clock time [zone] ?
```

The computer is asking you to type in the current time. First check if the time already entered is correct. The system time is displayed directly below the system date and they look like this:

```
System date is      12-Dec-86
Clock time is       14:50:21
```

Note that this is a *24-hour* clock, so 1 pm is hour 13, 2 pm is hour 14, and so on. Also note that the seconds are given. Thus the time shown in this example is 21 seconds past 14:50 (i.e., about 2:50 in the afternoon). Finally, note that below the system date and the system time is the time zone.

```
System date is      12-Dec-86
Clock time is       14:50:21
Time zone is         0: 0: 0
```

Set the time like this:

1. Type the time in the form **hh-mm-ss [tz]** where:
   - **hh** is one or two numbers representing the hour on a 24-hour clock
   - **mm** is one or two numbers representing the number of minutes past the hour
   - **ss** is one or two number representing the number of seconds past the minute (ignore this if you wish—it will automatically set to 00).
   - **tz** is a positive or negative number representing the number of hours to add to your local clock time to match the current GMT clock time. If you don't know what time zone to enter, do not enter any time zone.

   For example, if the time is 3:27 pm, you would type:

   ```
   15:27 [0]
   ```

   If You Make a Mistake... Simply press the [Back space] key until you erase your error. Then type the time over again. If you have already pressed [Return], then press it again (to enter null responses for date and time), and restart the Version command by pressing [V].

2. Press [Return] to register the new time. Note that the system time has now changed to the new value.

   If You Make a Mistake... Simply press the [V] key. The date prompt will reappear on the screen. Press [Return] to accept the current system date. The time prompt will then reappear. Type the time over again and press [Return].

You are now back at the Main Command Level. The time will be displayed until you press the space bar or execute a subsystem.

The clocks in HP Series 200 computers will need to be set every time power is applied. HP Series 300 computers have clocks that run on a battery and do not need to be set each time the computer is powered up. The time zone does need to be set each time however, and you will probably modify your AUTOSTART file to do so. See Chapter 10 before modifying your AUTOSTART file.

Notes about the Time Zone

Typical time zones values are positive for the Western Hemisphere and negative for the Eastern Hemisphere, as shown at the beginning of this chapter. If you do not know your time zone, consult a short-wave radio handbook or other suitable reference.

If you must specify a time zone that is not an "hours-only" difference between local time and GMT, you may specify the time zone just like specifying the time of day. For example,

```
11:21:30 [-3:30]
```

The -3:30 would indicate a time zone that is 3½ hours earlier than GMT.

Notes about Date and Time Preservation

Whenever the date, time or time zone are set (by the command interpreter's `V` command), the current values for date and time are set in the system clock, the battery-backed clock (if present), and the current system volume (if on-line and write enabled).

At system power up, the date and time are copied from the battery backed clock to the system clock. If a valid date has not been obtained, then the date and time will be read from the system volume. All this happens before the AUTOSTART/AUTOKEYS file is activated.

Pascal version 3.22 accurately keeps dates from January 1, 1970 to December 31, 2027. Previous versions of Pascal kept dates from March 1, 1900 to December 31, 1999.

Initializing Discs

All new discs—both flexible and hard—must be prepared for use by a process called **initialization** (sometimes called formatting).

This procedure describes how to initialize your discs with the Pascal Workstation.

CAUTION

Initializing destroys any data and programs stored on a disc. Be certain that you do not initialize a Pascal system disc, or any other disc containing valuable files.

If initializing a hard disc, do not interrupt the initialization process, as this may severely damage the drive.

Note

SRM Users: If your workstation is part of an Shared Resource Management (SRM) system, you can **not** initialize an SRM shared disc from your workstation. Your system manager initializes SRM shared discs from the SRM console using an SRM system command.

---

1. Make sure the following line appears across the top of the screen (press Stop if it doesn't):

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

2. Find the disc labeled ACCESS:.
3. Insert the ACCESS: disc into one of your disc drives.

4. Press the [X] key. The computer responds with:

```
Execute what file ?
```

5. Type:

```
ACCESS:MEDIAINIT [Return]
```

   MEDIAINIT is the name of the program that initializes discs.

6. Wait for MEDIAINIT's initial display:

```
Mediainit [Rev 3.22 1/15/87] 15-Feb-87 11:58:41

       Copyright 1987 Hewlett-Packard Company
              All rights reserved.

Volume ID ?
```

7. If initializing a flexible disc, make sure the disc is write-enabled (see "Write-protecting/enabling Flexible Discs" in Chapter 2), then insert the disc into a disc drive.

8. **Remove all *other* discs from your disc drives. Turn off all drives except the one containing the disc to be initialized.** This protects you from initializing the wrong disc by mistake. After initialization is completed, remember to turn all disc drives back on again, or unpredictable behavior may result.

9. At the bottom of MEDIAINIT's initial display, you are asked for the Volume ID (i.e., unit number) of the drive containing your disc. Look up the disc drive's unit number (the *first* unit number for hard discs) in the Pascal Workstation Worksheet (from the *Peripheral Installation Guide*). Then type:

   unit number [Return]

   where **unit number** is the unit number of the drive containing your disc.

   Flexible Disc Example: Suppose you look on your Pascal Workstation Worksheet and find that the unit number of the drive containing the disc to be initialized is `#3:`. Then you would type:

   `#3:` [Return]

Hard Disc Example: Suppose you look on your Pascal Workstation Worksheet and find that your hard disc has been assigned unit numbers 11 through 24. The first unit number is `#11:`, so you would type:

`#11:` Return

10. The computer will pause for several seconds, then finally display information about your disc on the screen. The last line reads:

```
Are you SURE you want to proceed? (Y/N)
```

Check that the unit number(s) given on the screen are correct for your disc drive, then press Y.

Note

If you are initializing a hard disc and you get an error message at this step, turn all of your hard discs off but the one you want to initialize. Then, with only this *one* hard disc turned on, reboot the Pascal Workstation. Begin again at step 1, and initialize your hard disc using unit number #11: (in step 9).

---

11. If you're using a double-sided micro disc drive, the following prompt will appear:

```
Formatting option? (defaults to 0)
```

Typical values are 0,1,2,3 or 4 and are explained below.

- If you are initializing a double-sided disc or high-density disc and want optimum performance for LIF, simply press Return to accept the default or type:

  1 Return

  This divides the disc into 256-byte sectors and gives a total storage capacity of 630 Kbytes (for double-sided discs) and 1.2 Mbytes (for high-density discs).

- If you are initializing a double-sided disc or high-density disc and want to divide the disc into 512-byte sectors, giving a total storage capacity of 710 Kbytes (for double-sided discs) and 1.4 Mbytes (for high-density discs), and will accept the associated performance degradation for usage with LIF, type:

  2 [Return]

- If you are initializing a double-sided disc or high-density disc and want to divide the disc into 1 024-byte sectors, giving a total storage capacity of 788 Kbytes (for double-sided discs formatted for storing HFS files) and 1.5 Mbytes (for high-density discs), and will accept the associated performance degradation, type:

  3 [Return]

- If you are initializing a *single*-sided, double-sided, or high-density disc for use in a single-sided disc drive, type:

  4 [Return]

  This divides the disc into 256-byte sectors and gives a total storage capacity of 270 Kbytes (for single- and double-sided discs) and 1.4 Mbytes (for high-density discs).

12. Wait for the prompt: `Interleave factor? [1..15] (defaults to 2)`
13. Look up your disc drive's interleave factor in the Pascal Workstation Worksheet. Then enter the interleave factor:

    - If no interleave factor is given on your worksheet, press [Return] to accept the default interleave factor.
    - If an interleave factor is given on your worksheet, type:

      interleave factor [Return]

      Where **interleave factor** is the interleave factor of your drive.

Example: If you look on your worksheet and find no interleave factor listed for your disc drive, press

[Return]

Another Example: If you look on your worksheet and find that the interleave factor for your disc drive is 3, type:

3 [Return]

Note

If you initialize a flexible disc in one model of disc drive and use it in another, disc access time may be degraded.

14. The screen now displays several messages. Wait for the message:

    ```
    Volume zeroing completed
    ```

    For flexible discs, this takes a few minutes. For large hard discs, this could take more than an hour.

Your disc is now ready for use. **Turn all of your disc drives on again** *unless* you have another disc to initialize. If you do, insert the new disc into the drive, press the [U] key to restart the initialization program, and go to step 9. You will need one initialized disc for each original Pascal Workstation disc in order to make a back-up copy of the Pascal Workstation in the next section.

Note

If you're initializing several discs, this process will take a while. Put a paper clip on this page and read the sections called "Understanding the File System" and "The Lessons of Disc Initialization" (found later in this chapter) while you wait.

Note

HFS Users: If you are planning to use the Hierarchical File System (HFS) directory structure, you will want to finish the initialization of your disc according to the instructions given in Appendix E of this manual.

Note

If you just initialized an HP 9133**A**, HP 9134**A** or HP 9135**A** hard disc drive (**not** option 010), see the special instructions in the next section.

Special Instructions for HP 9133A, 9134A and 9135A Drives

The "A" versions of the HP 9133, HP 9134 and HP 9135 disc drives (**not** option 010) may not be quite ready for use. You'll recall that step 12 of media initialization "zeroed" all volumes on the disc. In the case of the "A" drives, only the volume assigned to unit #11 was zeroed; if there are more volumes present, you must *manually* zero them. To do this:

1. Remain in the Filer and press the [Z] key to invoke the **Zero** command.
2. The computer displays the following prompt?

   ```
   Zero what volume ?
   ```

   Type:

   `#12:` [Return]

   to select the next unit number that could possibly be assigned to the disc.
3. The computer now asks:

   ```
   Number of directory entries ?
   ```

   Type:

   80 [Return]

   This will allocate 80 entries in this directory, allowing you to store up to 80 files in the volume.

If Something Goes Wrong...

Instead of this prompt, you may get:

```
Error: device absent or inaccessible
```

If this happens, skip to step 8.

---

4. The computer now asks:

   ```
   Number of bytes (1152000) ?
   ```

   This prompt is asking how large (in bytes) you want the volume to be. Press [Return] to accept the default value in parentheses.

5. The computer now asks:

   ```
   New directory name ?
   ```

   Type:

   V12: [Return]

   This is in keeping with the naming convention used by the media initialization program.

6. The computer now confirms your response:

   ```
   V12: correct ? (Y/N)
   ```

   Press [Y] to answer yes.

7. The computer now zeroes the volume and reports:

   ```
   Volume V12 zeroed
   ```

8. Repeat this procedure for all remaining volumes, substituting the next sequential unit number for **#12** in step 2, and naming the volume **VX:** in step 5, where **X** is the unit number you're zeroing. When you finally get the message:

   ```
   Error: device absent or inaccessible
   ```

   you have zeroed all volumes on your Winchester disc drive; it is now ready for use.

Understanding the File System

The disc initialization procedure introduced several terms that you may or may not have heard before — terms like *file, volume,* and *unit number.* Together these terms describe the Pascal Workstation's file system, which controls the way information is stored, retrieved, and transferred among the various parts of the system. This section how the file system is organized.

We begin with the most elemental packet of information: the *file.*

Files

Think of a file as a file folder that holds information. The information in a file might be a program, or a table of data, or it might even be this chapter of the *Pascal User's Guide.*

File folders are usually labeled so you can distinguish among them, and so are files. Each file has a **file name** associated with it that distinguishes it from every other file. On LIF discs, file names can be up to nine characters long, not including suffix. Names of files on SRM shared discs can be up to sixteen characters long, including suffix. Names of files on HFS discs can be up to fourteen characters long, including suffix.

File names can be composed of almost any character, but certain characters should be avoided because of their special significance to the Pascal system (see the File System chapter of the *Pascal Workstation System* manual for a list). For now, restrict your file names to the letters of the alphabet and the underscore (_), and you won't have any problems.

Volumes

File folders, of course, should be stored in a file drawer when not in use. Similarly, files are stored in **volumes.** A volume is analogous to a file drawer: it is a collection of files.

Volumes of files are stored on discs. Flexible discs, having only a relatively small amount of storage space, are big enough to hold only a single volume. Think of a flexible disc as a small filing cabinet with only one drawer (i.e., only one volume). Hard discs, however, have a much greater storage capacity and can hold several volumes if used with LIF. Imagine a hard disc as a large filing cabinet with several drawers, each drawer corresponding to one volume on the disc.

Volumes, like files, must have names so you can tell them apart. In the Pascal system, a volume name is usually called a **volume ID** (pronounced eye-dee for IDentifier). A volume ID is like a label on the front of a file drawer that distinguishes it from the other drawers in the filing cabinet. A well-chosen volume ID, like a well-chosen file drawer label, should suggest what is stored inside. LIF volume IDs can be up to six characters long. SRM volume IDs can be up to sixteen characters long.

A volume also has a **directory** associated with it. The directory keeps track of all files in the volume: what their names are, where they are located on the disc, how big they are, when they were last modified, and other such information. A directory on an HFS disc is similar but different. Appendix E describes the differences.

The directory is necessary because, unlike a well-managed filing cabinet, the Pascal system doesn't keep its files in alphabetical (or any other) order. It just searches for the first space in the volume big enough to hold the file, and slides it into the slot. If a real file drawer was "organized" this way, it would be nearly impossible to find anything--unless, of course, it had a directory.

The directory is the organized part of a disorganized volume. When the Pascal system stores a file in a volume, an entry is made for it in the directory. Later, when the file is retrieved, the disc drive uses the directory to find the requested file's location in the volume.

Again using the filing cabinet analogy, a directory would be a large list, posted on the front of a file drawer, showing the names, locations and other information about all files stored in that drawer. The file clerk would keep this list up-to-date and would use it to find a requested file.

Units

Now suppose every filing cabinet in the office is stacked one on top of another to form one tall tower of file drawers. Each file drawer in the tower is assigned a unique number. Actually, it is not the file drawer that is assigned the number, but the *location* in the tower where the file drawer is kept (i.e., the "slot" that the drawer slides into).

With this scheme, every file drawer is associated with a number that specifies its location in the tower. Since the numbers are attached to file drawer locations rather than to the file drawers themselves, you can remove one file drawer, replace it with another, and the new file drawer will now be associated with that number. Also note that a number may refer to an *empty* location that has no file drawer.

This situation exists in the Pascal system. Each *volume location* in the Pascal file system is called a **unit**. Each unit is assigned a unique **unit number** when the system is booted. Flexible disc drives can hold only one volume (i.e., one flexible disc) and are thus assigned one unit number. But Winchester discs can hold multiple volumes and thus are assigned one unit number for each volume they contain. Actually, the Pascal Workstation can assign more than one unit number to a flexible drive when desired, thus allowing a both LIF and HFS flexible discs to be used in the same drive.

Flexible discs drives have removable volumes—you can remove one disc and insert another (remember, each flexible disc holds one volume). A flexible disc volume is like a file drawer that can be removed from one slot and inserted into another. Winchester discs, however, do not have removable volumes, because the disc is not removable. A Winchester disc volume is like a file drawer that is permanently secured in its slot.

An Illustrative Example

This may sound complicated, so let's look at a real example. Refer to the diagram on the facing page.

Here we have two HP 9121D dual flexible disc drives and one HP 9134A Winchester disc drive. There are four *flexible* disc drives in the system, each capable of holding one volume (i.e., one disc). The *Winchester* disc is larger and is divided into four volumes. Thus, the total number of mass storage volumes in the system is:

(4 flexible disc drives × 1 volume each) + (1 hard disc × 4 volumes) = 8 volumes in system

Using the filing cabinet analogy, each HP 9121D dual flexible disc drive is a two-drawer filing cabinet, and the HP 9134A Winchester disc drive is a four-drawer filing cabinet.

In the illustration, we have stacked all filing cabinets into one tower and assigned unit numbers to each file drawer location. Notice that the drawers need not be assigned sequential unit numbers.

Unit #7 (drive 0 in the second HP 9121) contains no flexible disc volume (i.e., no file drawer) and thus has no volume ID associated with it. The unit number is still valid, however, because unit numbers identify locations where volumes may reside, *not* the volumes themselves. When this "empty drive" situation exists, the unit number is said to have no volume "on-line."

Flexible disc units 3, 4 and 8 *do* contain volumes, with volume IDs called MEMOS, PROGRAM 1 and REPORTS, respectively. Each volume ID is *associated* with the unit number of the drive where its volume resides. Understand, however, that this is a temporary association, and will be broken as soon as the flexible disc volume is removed from the unit. A flexible disc volume in a disc drive is said to be "on line."

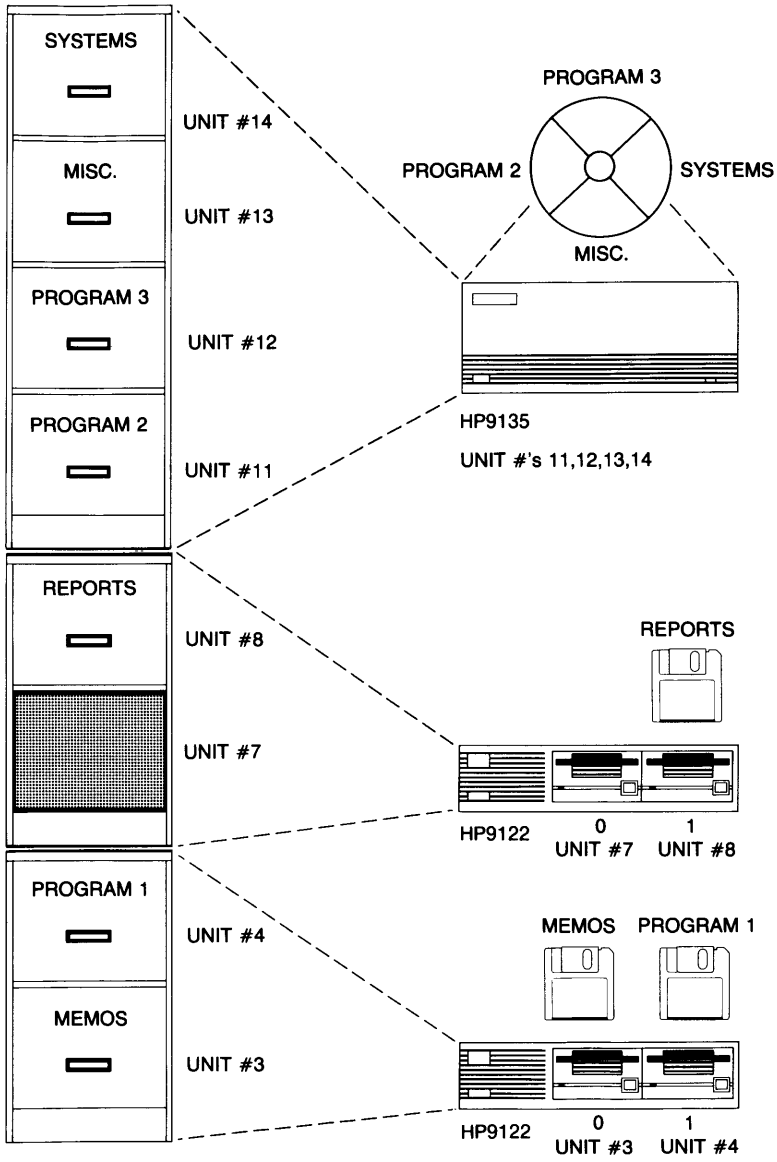
The four Winchester disc volumes–PROGRAM 2, PROGRAM 3, MISC. and SYSTEMS–are assigned unit numbers 11, 12, 13 and 14, respectively. Because the volumes on a Winchester disc are not removable, the volume ID of each volume is *permanently* associated with its unit number. (This is not quite true–you can rename a volume ID with one of the Pascal system's commands. But think of this as a purely superficial change, because the *contents* of the volume remain associated with the same unit number).

Note

The volume IDs shown in this example are for illustration purposes only. Most do not represent legal volume IDs.

If your workstation is part of an SRM system, see Appendix D for additional information regarding files, volumes, and units.

If you plan to use Hierarchical File System (HFS) discs, see Appendix E for additional information about HFS.


SYSTEMS
MISC.
PROGRAM 3
PROGRAM 2
REPORTS
PROGRAM 1
MEMOS
UNIT #14
UNIT #13
UNIT #12
UNIT #11
UNIT #8
UNIT #7
UNIT #4
UNIT #3
PROGRAM 3
PROGRAM 2
SYSTEMS
MISC.
HP9135
UNIT #'s 11,12,13,14
REPORTS
HP9122
0
UNIT #7
1
UNIT #8
MEMOS
PROGRAM 1
HP9122
0
UNIT #3
1
UNIT #4

The Lessons of Disc Initialization

By now you've initialized a few discs, and although you may not realize it, you have mastered many of the skills needed to use the Pascal Workstation. In this section, we take a closer look at the disc initialization procedure to learn how this task is representative of most that you'll be meeting along the way.

Reading the Command Line

The first thing you did before initializing your discs was check that the following line appeared at the top of the screen:

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

This is known as a **command line** because it contains a list of commands that are active at this level of the Pascal system. Notice that each command contains one upper case letter (e.g., **C** in Compiler, **X** in eXecute, etc.). This is the letter you must type to invoke the command.

The command line shown above gives the active commands for the Main Command Level—the top-most level of the Pascal system. The first word in this command line, **Command:**, identifies the Main Command Level. From here, you can branch into lower levels of the system, like the Filer and Editor, or invoke top-level commands, like **eXecute** and **Version**.

Recall that you restarted the media initialization program by pressing the [U] key. This should correspond to one of the commands, but notice there is no command in the command line that has an upper-case **U** in it. Where did this command come from? Press the [?] key to find out.

The **?** at the end of the command line indicates there are more active commands than can fit on one line. When you press [?], the rest of the commands are displayed:

```
Command: Assembler Debugger Memvol Newsysvol Permanent Stream User What ?
```

The **User** restart command is the second from the end. This is the command you invoked with the [U] key to restart the media initialization program.

Most of the Main Command Level is discussed in detail in the next chapter. What is important to remember here is that the top line of the screen will always tell you what level of the Pascal system you're in and what your command options are. Also remember that you need only press the upper-case letter in a command name to invoke it.

Selecting the Right Disc

The next step in initializing your discs was to insert the ACCESS: disc (i.e., the disc with the volume named ACCESS: on it) into your disc drive. You did this because the ACCESS: volume contains the file named MEDIAINIT.CODE, which in turn contains the program that initializes discs.

Note

Although you specified the file name as MEDIAINIT in the initialization procedure, its complete name is MEDIAINIT.CODE. The **eXecute** command automatically appends the file suffix .CODE, so there is not need to type it.

The Pascal system has several programs like MEDIAINIT.CODE that you will need to use from time to time. It's important to know which disc contains the program you want to use. You can then insert it into your disc drive where the Pascal system can access it. Appendix C contains a complete list of all files found on each disc of the Pascal system. Refer to it whenever you're wondering which disc to use.

One other point to make here: after you load a program from a disc into memory, you may remove the disc from the disc drive. The computer executes the copy of the program that resides in memory, and no longer refers to the copy on the disc. This is why you can remove the ACCESS: volume from the disc drive after running MEDIAINIT.CODE.

However, don't get the impression that once a program is loaded into memory, it is there to stay. Programs remain in memory only until another program is loaded. Then they are replaced by the new program, and must be reloaded before they can be re-executed.

Note

The Pascal system does allow you to permanently load programs into memory so they can't be "bumped" every time you load another program. This is discussed further in Chapter 5.

Specifying Files

Finally, the media initialization program also demonstrated how files should be specified. When the computer prompts you for a file, you must tell it two things: the *location* where the file is stored and what *file name* it is stored under.

The File Location

Remember that files are stored in volumes, so to specify the location of a file, you must specify the location of the volume that contains it. You may do this in either of two ways:

1. Give the *volume ID* of the volume containing the file, or
2. Give the *unit number* that is associated with the volume containing the file.

When you specify the location of a file with the volume ID, the computer searches all of its disc drives for a volume with the correct volume ID. If you specify the location with a unit number, the computer looks *only* at the disc that currently resides in the specified unit.

The File Name

The file name is just the name of the file where the program is stored.

Putting It All Together

To completely specify the file, simply append the file name to the file location, remembering that the file location can be either a volume ID or a unit number. A colon (:) must separate the file location from the name.

Note

The colon is usually considered part of the file location; i.e., both the volume ID and the unit number include the colon.

The **eXecute** command asked you to specify the file containing the program you wanted to run with the prompt:

```
Execute what file ?
```

You responded with:

```
ACCESS:MEDIAINIT
```

ACCESS: is the volume ID of the volume containing the file named MEDIAINIT.CODE. You could just as easily have specified the unit number of the disc drive containing the ACCESS: volume instead of the volume ID. For example, if the disc drive containing the ACCESS: volume was assigned unit #3:, you could have typed the following line instead:

```
#3:MEDIAINIT
```

Later in this chapter, you will discover which unit number(s) the Pascal system has assigned to each disc drive in your system.

Specifying Hierarchical Files

To specify a file on an SRM (Shared Resource Manager) disc or an HFS (Hierarchical File System) disc, a little more information is usually needed. Both SRM and HFS discs allow a directory to contain not only files but other directories as well. This means that a file may exist within a directory that appears within another directory.

Recall the file cabinet analogy described earlier, where the file clerk posted a list of the drawer's contents on the front of each drawer. He may now want to keep a master list that shows the names of all these other lists. By looking at the master list, he would then know how to find a particular drawer without having to look at the list on each drawer. This is the hierarchy.

To specify a file on a hierarchically structured system, each directory leading to the file is concatenated with a slash character (/). For example if there was a volume called PASCAL: and within that volume there was a directory called ACCESS, and within that directory there was a file called MEDIAINIT, you could type:

```
PASCAL:ACCESS/MEDIAINIT
```

or

```
#11:ACCESS/MEDIAINIT
```

Remember, this only applies to hierarchically organized file systems. See Appendix D for complete examples of specifying SRM files; Appendix E for HFS.

Responding to Prompts

The media initialization process exposes you to a number of **prompts.** Prompts are nothing more than questions that appear on the screen. The computer waits for you to respond to a prompt before it resumes what it's doing.

Prompts are distinguished by the ways you respond to them and by the "clues" they provide. Let's examine each type:

Straight Questions

The first type of prompt you saw was a straight question that appeared just after the media initialization program began running:

```
Volume ID?
```

You responded to this prompt by typing the unit number of the disc drive containing the disc you wanted to initialize.

Note

Don't let the fact that the prompt asks for the volume ID confuse you. Since the disc is not initialized, it contains no volume yet, and thus has no volume ID. The prompt would be more accurate if it asked for the unit number of the drive containing the uninitialized disc.

To signal the computer that you were finished typing your response, you pressed [Return]. In general, if your response requires more than a single keystroke, you will finish by pressing [Return].

Yes/No Prompts

The next prompt MEDIAINIT displayed was:

```
Are you SURE you want to proceed ? (Y/N)
```

This is a good example of a Yes/No prompt. It requires only a single keystroke: [Y] for Yes or [N] for no. You do not need to follow your response with [Return]. These prompts often appear when you're performing a potentially dangerous operation, giving you the opportunity for a last minute retreat. Media initialization is potentially dangerous because it destroys the existing volume(s) on a disc. If the disc is new, this is no problem, but if it is old and contains valuable data, it could be a devastating loss.

Value Range Prompts

A value range prompt tells you what range of values it will accept as a valid response. The range of values is usually enclosed within square brackets `[ ]`. The following prompt is of this variety:

```
Interleave factor? [1..15] (defaults to 1)
```

This prompt will accept any response in the range of 1 to 15.

Prompts with Default Values

The preceding prompt is also a good example of a prompt with a default value. A default value is the value the prompt will "assume" if no value is specified. Thus, when you respond to this prompt by simply pressing [Return], the default value is taken as your answer.

Default values are usually offered to suggest what the computer perceives as the best or most likely answer. Unless you know something the computer doesn't, it is usually best to accept the default.

Aborting a Prompt

If you ever reach a prompt by mistake and simply want to abort the operation, just press [Return] without giving an answer. The prompt will usually disappear.

For example, if you restarted the media initialization program by mistake and wanted to abort it, you could simply press [Return] in response to the prompt:

```
Volume ID ?
```

Notice, however, that this won't work with prompts that have default values, such as the last prompt in the media initialization program:

```
Interleave factor? [1..15] (defaults to 1)
```

Pressing [Return] in response to this prompt is a legitimate answer. It says that you want to accept the default value as your response. To abort this kind of prompt, you must press [Stop]. As in any other situation, [Stop] terminates the current operation immediately and returns you to the Main Command Level.

Verifying Unit Number Assignments

In the *Peripheral Installation Guide*, you completed the Pascal Workstation Worksheet showing how unit numbers are assigned in your particular computer system. In this section, you will verify that these assignments are correct.

The Pascal system may not recognize all of your disc drives and other peripherals if your system contains:

- More than one 30-megabyte (or larger) Winchester disc drive
- More than 3 *dual* flexible disc drive units
- More than one SRM shared disc
- A serial (RS-232C) printer
- An HP 98259A Bubble Memory card
- An HP 98255A EPROM card

To make the Pascal system recognize these situations, you must modify the way Pascal searches for peripherals. See the Special Configurations chapter of the *Pascal Workstation System* manual for details. For the moment, just use the devices that the system can recognize.

Verifying Flexible Disc Drive Units

To find out how unit numbers were assigned to your flexible discs, follow this procedure:

1. Turn all flexible disc drives on.
2. Insert the ACCESS: disc into any one of your flexible disc drives.
3. Make sure the command line for the Main Command Level appears across the top of the screen (press [Stop] if it doesn't):

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

4. Press [F] to load the Filer.
5. Press [V] to execute the **Vols** (Volumes) command. This command displays a list of all volumes that are on-line, along with their associated unit numbers.

The list looks something like this:

```
Volumes on-line
  1    CONSOLE:
  2    SYSTERM:
  3 *  ACCESS:
  6    PRINTER:
Prefix is - ACCESS:
```

The left-hand column of numbers gives the unit numbers of all on-line devices. The right-hand column gives the volume ID associated with each unit number.

6. Find the volume ID `ACCESS:` in the list. The corresponding unit number is the unit number assigned to the flexible disc drive where the ACCESS: disc resides (#3 in the example above). Remember that this unit number is not permanently assigned to the ACCESS: volume, but *is* permanently assigned to the drive in which ACCESS: resides.

Note

Again, we must hedge on one statement. It's true that unit numbers are permanently assigned to a disc drive—provided you insert the boot disc into the same disc drive every time you boot the system. If you boot from a different flexible disc drive, or from a Winchester disc, flexible disc unit number assignments may change. This is because flexible disc drive unit numbers are assigned *relative* to the drive containing the boot disc.

---

7. Check that the Pascal Workstation Worksheet lists the correct unit number for this drive. If it doesn't, update the worksheet with the correct information.
8. Now insert the ACCESS: disc into another flexible disc drive. Press [V] again and find the ACCESS: volume in the list. Check the Pascal Workstation Worksheet for accuracy. Repeat this step until you have verified the unit numbers for all flexible disc drives in your system.

Note

The Pascal system will recognize a maximum of three *dual* flexible disc drive units.

Verifying Winchester (Hard) Disc Drive Units

As mentioned earlier, Winchester (hard) discs initialized by the Pascal system may contain more than one volume due to their size (if set up for LIF). This procedure will tell you which unit numbers are assigned to a particular drive.

1. Make sure you are in the Filer; the word `Filer:` should appear as the first word in the command line at the top of the screen. If you are not in the Filer, follow the first two steps of the preceding procedure, "Verifying Flexible Disc Drive Units," then come right back.
2. Turn *off* all hard disc drives except one.
3. Press [V] to invoke the **Vols** command. All volumes that are on-line are listed on the screen. The list looks something like this:

```
Volumes on-line
  1   CONSOLE:
  2   SYSTERM:
  6   PRINTER:
 11 * V11:
 12 # V12:
 13 # V13:
 14 # V14:
 15 # V15:
 16 # V16:
 17 # V17:
 18 # V18:
 19 # V19:
 20 # V20:
 21 # V21:
 22 # V22:
 23 # V23:
 24 # V24:
Prefix is - V11:
```

The left-hand column gives all of the unit numbers for each volume; the right-hand column gives the volume ID associated with each unit number.

All unit numbers listed on the screen that fall within the range 11 through 40 are assigned to the hard disc that is turned on. Each unit number corresponds to one volume on the disc. The initialization process assigns volume IDs to each of the volumes, in the form **vX:**, where **X** is the unit number assigned to that volume.

Note

If no unit numbers in the range 11 through 40 appear in the list, or if fewer unit numbers have been assigned to the disc than you expected, see the Special Configurations chapter of the *Pascal Workstation System* manual.

---

4. Check that the *Perpherial Installation Guide's* Pascal Workstation Worksheet lists the correct unit numbers for this drive. If it doesn't, update the worksheet with the correct information.
5. Press [Q] to Quit the Filer.
6. Turn your other Winchester discs back on. Allow them to spin-up, then press [I] to Initialize the unit table.
7. If you have another winchester disc, press [F] to re-enter the Filer and repeat this procedure.

Verifying Other Discs

SRM System Discs

If you are using an SRM system, follow this procedure to see which units are assigned to the SRM system. Also see Appendix D for more information regarding SRM systems.

1. Make sure you are in the Filer; the word `Filer:` should appear as the first word in the command line at the top of the screen. If you are not in the Filer, follow the first two steps of the preceding procedure, "Verifying Flexible Disc Drive Units," then come right back.

2. Press [V] to invoke the **Vols** command. All volumes that are on-line are listed on the screen. The list looks something like this:

```
Volumes on-line
  1   CONSOLE:
  2   SYSTERM:
  5 # MYSRM:
 45 * SYSTEM21:
Prefix is - MYSRM:
```

The left-hand column gives all of the unit numbers for each volume; the right-hand column gives the volume ID associated with each unit number.

The SRM appears as two units. Unit number 5 indicates the current working directory and unit number 45 indicates the system directory.

HFS Discs

If you are using an Hierarchical File System (HFS) disc, it appears to the Pascal Workstation as a single volume. The default unit number is the same as a Winchester disc (#11:).

```
Volumes on-line
  1   CONSOLE:
  2   SYSTERM:
  3 # BOOT:
  4 # ACCESS:
 11 * HFS:
Prefix is - HFS:
```

The left-hand column gives all of the unit numbers for each volume; the right-hand column gives the volume ID associated with each unit number.

If you booted from an HFS hard disc, unit #46: will be the system directory.

Making a Back-up Copy of the Pascal Workstation

Description

Protect your software investment by making a back-up copy of the Pascal Workstation. Then if your working copy is destroyed, you can always make another.

You will need one blank, initialized disc for each original Pascal Workstation disc.

Two procedures are provided:

- Use Procedure 1 if you have two flexible disc drives.
- Use Procedure 2 if you have only one flexible disc drive.

Procedure 1:
Copying Discs with Two Drives

First we will copy the ACCESS: disc.

1. Find the ACCESS: disc. Make sure it is **write-protected** (see "Write-enabling/protecting Flexible Discs" in Chapter 2), then insert it into a flexible disc drive. Look up the unit number of this drive in the Pascal Workstation Worksheet. This is your **source unit number**.
2. Insert a blank, initialized flexible disc into your other flexible disc drive. Look up the unit number of this drive in the Pascal Workstation Worksheet. This is your **destination unit number**.
3. Press [F] to load the Filer from the ACCESS: disc.
4. Wait for the following line to appear across the top of the screen:

```
Filer: Change Get Ldir New Quit Remove Save Translate Vols What Access Udir ?
```

5. Press [F] to invoke the **Filecopy** command.
6. Wait for the following prompt to appear:

   ```
   Filecopy what file ?
   ```

7. Type:

   source unit number [Return]

   where **source unit number** is the unit number of the drive containing the ACCESS: disc.

   Example: Suppose you look on the Pascal Workstation Worksheet and find that the unit number of the drive containing the ACCESS: disc is `#3:`. Then you would type:

   `#3:` [Return]

8. The computer now responds with the prompt:

   ```
   Filecopy to what ?
   ```

9. Type:

   destination unit number [Return]

   where **destination unit number** is the unit number of the drive containing the blank, initialized disc.

   Example: Suppose you look on the Pascal Workstation Worksheet and find that the unit number of the drive containing the blank, initialized disc is `#4:`. Then you would type:

   `#4:` [Return]

10. The copying process now commences, as the contents of the source disc is read into your computer's memory. When finished reading, the following prompt appears:

    `Destroy directory VX ? (Y/N)`

    where **VX** is the directory name given to the disc when it was initialized (**X** is the unit number of the drive in which it was initialized).

11. Type your response to the prompt:

    - If the directory name displayed is the same as one of your Pascal system discs (e.g., ACCESS:, BOOT2:, etc), press [N] to abort the operation immediately! Start over, making sure you insert the proper disc into the proper drive.
    - If the directory name displayed is V3, V4 or something similar, press [Y] to complete the copying process.

12. The source disc is now copied to the destination disc. When finished, the computer displays a line like:

    `ACCESS:  ==> #4:`

13. The ACCESS: disc has now been copied to the blank, initialized disc. Remove the original ACCESS: disc and the new duplicate ACCESS: disc from their disc drives. Label the duplicate disc with the same information that appears on the original disc.

14. Insert another **write-protected** Pascal Workstation disc and another blank, initialized disc into your disc drives. Starting again at step 5, repeat this procedure for each Pascal Workstation disc, until you have a complete set of duplicate discs. Then store your original (master) copies in a safe place. Use your duplicate discs for day-to-day operation.

When finished, press [Q] to quit the Filer program.

Procedure 2:
Copying Discs with One Drive

First we will copy the ACCESS: disc.

1. Find the ACCESS: disc. Make sure it is **write-protected** (see "Write-enabling/protecting Flexible Discs" in Chapter 2), then insert it into a flexible disc drive.
2. Press [F] to load the Filer from the ACCESS: disc.
3. Wait for the following line to appear across the top of the screen:

```
Filer: Change Get Ldir New Quit Remove Save Translate Vols What Access Udir ?
```

4. Press [F] to invoke the **Filecopy** command.
5. Wait for the following prompt to appear:

   ```
   Filecopy what file ?
   ```

6. Type:

   ```
   #3: Return
   ```

7. The computer now responds with the prompt:

   ```
   Filecopy to what ?
   ```

8. Type:

   ```
   #3: Return
   ```

9. The copying process now commences, as the contents of the disc is read into your computer's memory. When finished reading, the following prompt appears:

   ```
   Please mount DESTINATION in unit #3
   'C' continues <esc> aborts
   ```

10. Remove the ACCESS: disc and insert a blank, initialized disc.
11. Press [C] to continue copying.

12. The computer now displays:

```
Destroy directory V3 ? (Y/N)
```

13. Type your response to the prompt:

    - If the directory name displayed is the same as one of your Pascal system discs (e.g., ACCESS:, BOOT2:, etc), press [N] to abort the operation immediately! Start over, making sure you replace the original disc with the blank, initialized disc in step 10.
    - If the directory name displayed is V3, press [Y] to complete the copying process.

      This may take more than one pass. For example, when copying a double-sided disc with only 512 Kbytes of memory in the computer.

14. When finished, the computer displays:

```
ACCESS:                              ==> #3:
```

15. The ACCESS: disc has now been copied to the blank, initialized disc. Remove the new duplicate ACCESS: disc from the disc drive. Label the duplicate disc with the same information that appears on the original disc.
16. Insert another **write-protected** Pascal Workstation disc into your disc drive. Starting again at step 4, repeat this procedure for each Pascal Workstation disc, until you have a complete set of duplicate discs. Then store your original (master) copies in a safe place. Use your duplicate discs for day-to-day operation.

When finished, press [Q] to quit the Filer program. Be sure to write- enable your working copy of SYSVOL:.

Notes:

Using the Keyboard and Display 4

Now that the preliminaries of Chapter 3 are out of the way, it's time to look at the ways the Pascal system defines your computer's screen and keyboard. In this chapter, you will:

- Learn how to use the type-ahead buffer
- Learn how to read status information from the screen
- Become familiar with the most important keys on the keyboard

Getting Ready for Chapter 4

This chapter contains numerous demonstrations involving the Filer and Editor **subsystems**. A subsystem is a discrete functional part of the Pascal Workstation that handles a major programming task. The Editor handles editing functions; the Filer handles filing operations.

Before you begin working through the demonstrations, you should know how to enter and exit the Filer and Editor. In all cases, you enter from, and exit to, the Main Command Level.

Entering the Editor

Follow this procedure to enter the Editor:

1. Make sure the command line for the Main Command Level appears on the screen:

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

   Press [Stop] if it doesn't.

   If your workstation is part of an SRM system or if the Pascal system has already been installed on a hard disc, skip to step 3.

2. Insert your working copy of the ACCESS: volume into one of your flexible disc drives.
3. Now press the [E] key to load and run the Editor subsystem. If you inserted the ACCESS: volume into one of your flexible disc drives, leave it there while reading this chapter.
4. The following display should now appear on the screen:

```
Editor  [Rev 3.2   15-Jan-87]

Copyright 1987 Hewlett-Packard Company.
        All rights reserved.

No workfile found.
File? (<ret> for new file, <stop> exits)
:
```

   This display is the "front door" of the Editor; it is asking you if there is an existing file that you want to edit. Since you have none, simply press [Return] to indicate that you want to create a new file. The Editor's command line then appears on the screen:

```
Edit: Adjst Cpy Dlete Find Insrt Jmp Rplace Quit Xchng Zap ?
```

You are now inside the Editor.

Exiting the Editor

Use the following procedure to exit the Editor:

1. Make sure the Editor's command line appears on the screen:

```
Edit: Adjst Cpy Dlete Find Insrt Jmp Rplace Quit Xchng Zap ?
```

   If it doesn't, press [Return], [Select] or [ESC].

2. Press [Q] to quit the Editor.

3. The following prompt now appears on the screen:

```
>Quit:
     Update the workfile and leave
     Exit without updating
     Return to the editor without updating
     Write to a file name and return
```

   Since you will create no text of any value in the demonstrations, press [E]. This corresponds to the option, **Exit without updating.**

4. If you have not inserted text since you entered the Editor, you are now back at the Main Command Level. However, if you have inserted text, the Editor gives you one last chance to preserve your golden words before they are lost forever. It asks:

```
Are you sure you want to exit without updating?
     Type Yes  to Exit Without Update
     Type No   to Return to Editor
```

   Well, yes, you're sure you don't want to save them, so press [Y].

You are now back at the Main Command Level.

Entering the Filer

Follow this procedure to enter the Filer:

1. Make sure the command line for the Main Command Level appears on the screen:

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

   Press [Stop] if it doesn't.

   If your workstation is part of an SRM system or if the Pascal system has already been installed on a hard disc, skip to step 3.

2. Insert your working copy of the ACCESS: volume into one of your flexible disc drives.

3. Now press the [F] key to load and run the Filer subsystem. If you inserted the ACCESS: volume into one of your flexible disc drives, leave it there while reading this chapter.

4. The following command line should now appear at the top of the screen:

```
Filer: Change Get Ldir New Quit Remove Save Translate Vols What Access Udir ?
```

You are now inside the Filer.

Exiting the Filer

Use the following procedure to exit the Filer:

1. Make sure the Filer's command line appears on the screen:

```
Filer: Change Get Ldir New Quit Remove Save Translate Vols What Access Udir ?
```

   If it doesn't, press [Return] or [N].

2. Press [Q] to quit the Filer.

You are now back at the Main Command Level.

Pascal Screen Organization

Every language or operating system organizes your computer's screen a little differently. In this section, we examine the assortment of information that the Pascal system displays on your screen.

The Command Line

You have already seen several examples of command lines—those strings of commands that run across the top of the screen. Command lines tell you what commands are active, what operational mode the computer is in, what special keys are active in the current mode, how to exit the current mode, and so on.

The operational mode of the computer is always identified by the first word in the command line. Try this demonstration and watch the first word change:

1. Make sure the Pascal system is at the Main Command Level (press [Stop] if it isn't). Notice that the first word is `Command:`; this indicates you are at the Main Command Level.
2. Enter the Editor as described earlier in this chapter. The first word in the command line now changes to the current mode, i.e., `Editor:`.
3. Now press [I] to enter insert mode. Notice that the first word is now `Insert:`.

While in insert mode, look at the rest of the information in the command line:

```
Text <bs>, <clr ln>  [<sel> accepts, <esc> escapes]
```

- `Text` tells you that you can type in text while in insert mode.
- `<bs>, <clr ln>` tell you that the [Back space] and [Clear line] keys are active in insert mode. Whenever you see something enclosed in angle brackets (<>), it signifies an abbreviation for a key.

Note

If you don't have an HP HIL keyboard, such as the 46020A, key abbreviations that appear on the screen will be different. `<exc>` replaces `<sel>`, `<sh-exc>` replaces `<esc>`, and `<ent>` replaces `<ret>`. See the "Exiting from Commands" procedure in Appendix A.

The rest of the line is especially important, for it tells you the various ways to exit insert mode:

- `<sel>` represents the [Select] key on the HP 4602xA keyboard. The line says that `<sel> accepts`, which means that if you're satisfied with the text you've entered in insert mode, you can press [Select] to accept it (i.e., make it permanent).
- `<esc>` represents the [ESC] key (or you can also press [Shift] [Select]). If you don't want to keep the text you've typed in insert mode, you can press [ESC] to escape; i.e., anything typed during your last insert session will be erased.

Try it. Type in something, and then press [Select] to accept it. Notice that it is permanent; you can use the arrow keys to move anywhere along the line. Now press [I] to enter insert mode again, and type a few more words. Press [ESC] (or [Shift]-[Select]) to exit insert mode, and observe that your latest words are erased.

Note

If you have a bit-mapped display, the Editor's command line may disappear if you scroll vertically. Press the [V] key to retore it.

Exit the Editor as described earlier in this chapter.

Status Indicator

The lower-right corner of the screen is reserved for the status indicator. The status indicator tells you what the computer is doing at any given moment.

The most common status indicator symbol you'll see is I/O. This is displayed whenever the computer is waiting for you to type something at the keyboard, such as a response to a prompt or a command from the command line. Look at the status indicator now—it's probably displaying the I/O symbol.

If you're loading one of the Pascal subsystems, such as the Filer, Editor or Compiler, the status indicator will display the first letter of the subsystem while it is being loaded. For example, make sure the Pascal system is at the Main Command Level (press [Stop] if it isn't). Load the Filer as described earlier in this chapter. Notice that the status indicator changes to `F` while the Filer is being loaded. Exciting stuff, right? Exit the Filer as described earlier in this chapter.

If you're running an application program, either an `X` (for eXecute), an `R` (for Run) or a `U` (for User restart) is displayed.

If the Debugger is loaded and you press [Break], a `P` status indicator will be displayed and program execution will be suspended. If your ever find yourself unable to execute commands from the keyboard, check for this symbol. Press [f4] (**CONT**) to resume operation.

Type-ahead Buffer

The type-ahead buffer allows you to issue commands to the computer faster than the computer can carry them out. Instead of issuing one command, waiting for it to complete, then issuing another command, you can type in a whole string of commands and then wait for the computer to catch up.

To conveniently demonstrate the type-ahead buffer, you must first execute the Debugger subsystem. Pressing the [Break] key when the Debugger is present prevents the computer from executing keyboard commands; instead, all keystrokes are entered into the type-ahead buffer.

To execute the Debugger, insert your working copy of the ASM: volume into one of your flexible disc drives. At the Main Command Level, press [X] to invoke the **eXecute** command, and type the following line in response to the prompt (don't forget the period):

```
ASM:DEBUGGER. [Return]
```

When the command line for the Main Command Level reappears, the Debugger has been loaded.

Now press [Break]. Notice the status indicator has changed to `p`, telling you that the system has paused. Carefully type the following line:

```
FVQE [Return] I [Return] HI! [Select]
```

Note

If you can't see the letters you're typing along the bottom of the screen, press [Menu] to turn off the softkey labels.

Notice the string of characters at the bottom of the screen. These are your keystrokes sitting in the type-ahead buffer, waiting to be executed. Press [f4] to resume execution from the keyboard (if you do not have the HP 4602xA keyboard, look up **CONT** in the Key Correspondence Table and press the appropriate key). The computer begins picking out keystrokes from the type-ahead buffer and executing them.

Type:

```
QEY
```

to exit the Editor.

You can delete one character from the type-ahead buffer by holding down [CTRL] and pressing [Back space]. This is handy if you mistype a command, provided you can delete the errant character before the computer fetches it. Don't type [Back space] by itself and expect it to delete the last command; this only enters a back-space *character* into the buffer.

To demonstrate this, press [Break] again, and type a few letters. Then hold down [CTRL] and press [Back space] a few times and watch the characters disappear from the end of the type-ahead buffer. Delete all of your keystrokes in this manner, then press [f4] (**CONT**).

You can also clear the entire type-ahead buffer by holding down [CTRL] and pressing [Clear line], should you want to abort all of your type-ahead commands.

To demonstrate this, press [Break] and type a few letters. Then hold down [CTRL] and press [Clear line], and see how the entire buffer is cleared. Press [f4] (**CONT**) to resume normal operation.

HP-HIL Devices

HP-HIL devices, which are only available on Model 217, 237 and Series 300 Computers, are connected to the computer through the HIL (Human Interface Loop) card. This section describes the brief software installation procedure that is required in order to use HP-HIL devices as cursor-movement input devices, and lists the input capabilities that the devices provide for the Pascal Workstation. Use of HP-HIL graphics input devices is described in the "Interactive Graphics" chapter of the *Pascal Graphics Techniques* manual.

Note

It is inadvisable to connect or disconnect HP-HIL devices while the system is running, as this may hang the system. The *only* times when connecting and disconnecting HP-HIL devices is safe is when the system's power is off, or when the system's Command Interpreter prompt is being displayed *and* the time/date display (from the **Version** command) is *not* being displayed.

Installing the Mouse Drivers

Two drivers provide support of using the mouse with Pascal. They are both on the CONFIG: disc shipped with your system (if your Pascal Workstation came on double-sided micro discs, those drivers are on the ACCESS: disc). There are two ways that you can install them in the system.

- You can eXecute these driver files, and they "permanently load" themselves into memory; that is, they remain in memory until the system is re-booted. (This section describes this installation method.)
- You can add these driver files (modules) to the INITLIB file, which is automatically loaded into memory during the booting process. See "Adding Modules to INITLIB" in the "Special Configurations" chapter of the *Pascal Workstation System* manual for further details.

To load the HP HIL module, make sure that the CONFIG: (or ACCESS:) volume is on-line and type the following (don't forget the period in the file name):

- [X] `CONFIG:HPHIL.` [Return] if you have single-sided micro discs
- [X] `ACCESS:HPHIL.` [Return] if you have double-sided micro discs

After the program file is loaded, the system executes it. When the driver program has finished its execution, you can load the next module—MOUSE.

Make sure that the CONFIG: (or ACCESS:) volume is on-line and type the following (don't forget the period):

- [X] `CONFIG:MOUSE.` [Return] if you have single-sided micro discs
- [X] `ACCESS:MOUSE.` [Return] if you have double-sided micro discs

When this driver program has finished its execution, you can begin using the mouse.

Note that if you are using the mouse with the Device–Independent Graphics Library (DGL), you may obtain better mouse performance by installing the `DGL_REL` mouse driver. See the *Pascal Workstation Manual* and the *Pascal Graphics Techniques Manual* for details concerning the mouse and DGL.

Mouse Capabilities

The HPHIL and MOUSE modules provide the following capabilities:

- Moving the mouse right and left will move the alpha screen cursor right and left, respectively.
- Moving the mouse forward and backward will move the alpha screen cursor up and down, respectively.
- For a **two-button** mouse:
  - Pressing the right mouse button is identical to pressing the [Return] key.
  - Pressing the left mouse button is identical to pressing the [Select] key.
- For a **three-button** mouse:
- Pressing the right mouse button produces an ASCII character 30.
  - Pressing the middle mouse button is identical to pressing the [Return] key.
  - Pressing the left mouse button is identical to pressing the [Select] key.
- More details of moving the cursor and using the [Return] and [Select] keys are given in the subsequent "Using the 46020 Keyboard" section.

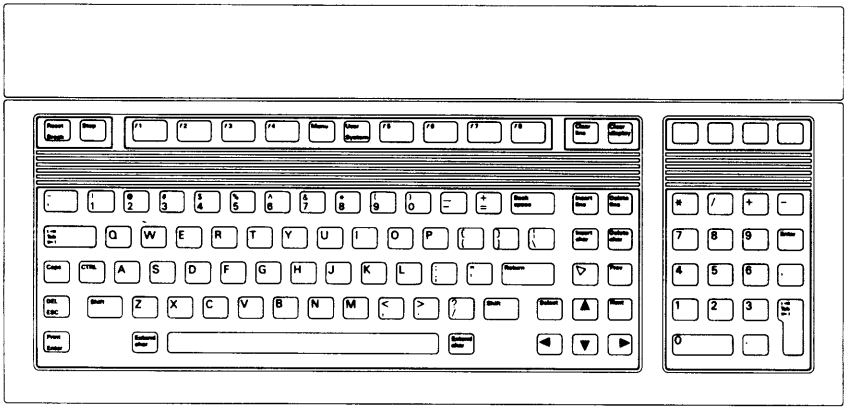
Installing Other "Mouse-like" HP-HIL Device Drivers

To use all other "mouse-like" HP-HIL devices supported by the Pascal Workstation (e.g., the Knob), you need only load the HPHIL and MOUSE modules as you do for the mouse. See the preceding section, "Installing the Mouse Drivers " for instructions.

Keyboards

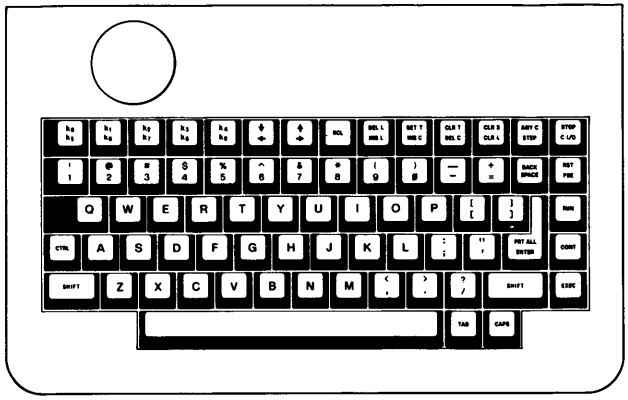
As you have seen in previous chapters, most Pascal system commands are issued with a single keystroke, and use only regular typewriter keys. Because of the simplicity and power of this scheme, many of the special function keys on your keyboard are not needed, and thus, not defined.

The following sections demonstrate the three available Series 200/300 computer keyboards as defined by the Pascal Workstation. Match your keyboard to the following diagrams, then turn to the indicated page and work through the demonstrations.



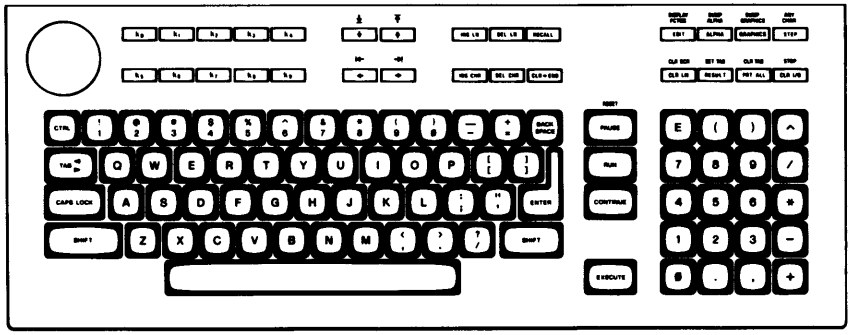
HP 46020A and 46021A Keyboards

Turn to page 72



HP 98203A Keyboard

Turn to page 103



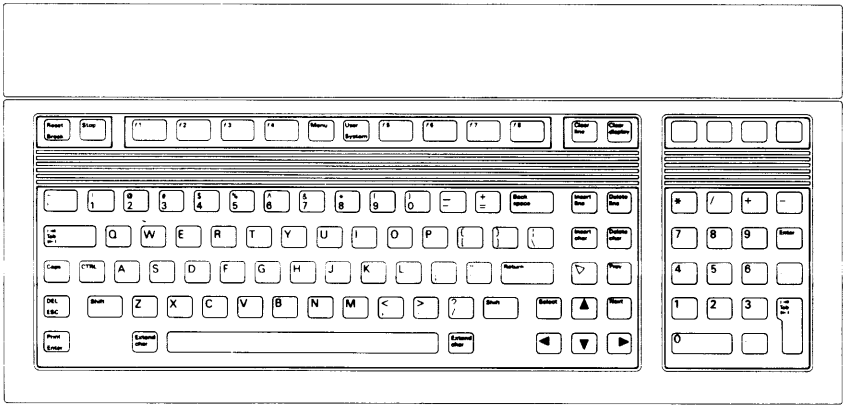
HP 98203B/C Keyboards

(Model 226/236 also)

Turn to page 90

HP 46020A and HP 46021A Keyboards

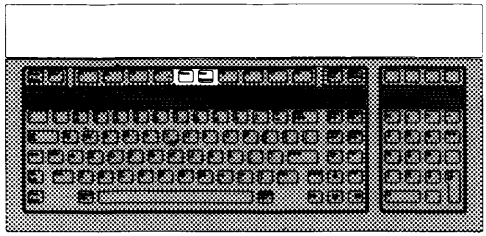
This section describes and demonstrates all of the keys the Pascal system defines on the HP 46020A keyboard.



Note

The "cursor" referred to in the following demonstrations is the underline character that appears on the screen. The cursor marks the position where the next typed character will be displayed.

Modes of Operation



The keyboard has two modes of operation: **system mode** and **user mode**. The mode is set with the [System]/[User] key.

The mode of operation defines the functions of the keys [f1] through [f8] that run across the top of the keyboard. In system mode, these keys are given definitions by the Pascal system, and perform such functions as display control, printing a display, etc. In user mode, the keys are given definitions that you (or some application program) provide.

The keyboard is set to system mode after the Pascal system is booted. System mode is designated by an `S` in the lower-right corner of the screen, immediately to the left of the status indicator.

User mode is entered by pressing [User]. User mode is designated by a `U` in the lower-right corner of the screen.

The definitions of the keys [f1] through [f8] can be displayed along the bottom of the screen with the [Menu] key. The functions of the [System], [User] and [Menu] keys are described below.

[System] turns on system mode, giving keys [f1] through [f8] (and [Shift]-[f1] through [Shift]-[f8]) their Pascal system definitions (see the "System-defined Keys" section that follows).

[User] turns on the user mode, giving keys [f1] through [f8] definitions provided by an application program (see the "User-defined Keys" section that follows).

[Menu]

The actual definitions (labels) for the system-defined and user-defined keys can be displayed at the bottom of the screen. [Menu] and [Shift]-[Menu] are used to control the display of these labels. [Menu] displays the un-shifted definitions of keys [f1] through [f8]; i.e., the definitions of these keys when pressed by themselves. [Shift]-[Menu] displays the shifted definitions of keys [f1] through [f8]; i.e., the definitions of these keys when pressed in conjunction with the [Shift] key.

The actions of [Menu] and [Shift]-[Menu] depend on the current "state" and mode of the keyboard.

When in system mode, possible actions of [Menu] are:

- If the the *un-shifted* labels are currently displayed, they are turned off.
- If no labels are currently displayed, the *un-shifted* labels are turned on.
- If the *shifted* labels are currently displayed, they are replaced by the *un-shifted* labels.

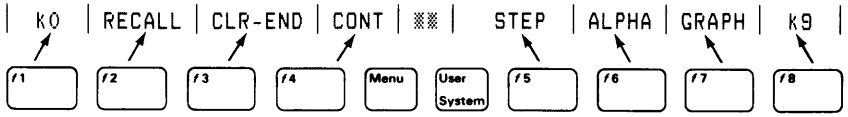
When in system mode, possible actions of [Shift]-[Menu] are:

- If the *un-shifted* labels are currently displayed, they are replaced by the *shifted* labels.
- If no labels are currently displayed, the *shifted* labels are turned on.
- If the *shifted* labels are currently displayed, they are turned off.

This will sound confusing until you see it work. First, observe how the [System] and [User] keys work. If an `S` is not displayed in the lower-right corner of the screen, press [System] to put the keyboard in system mode. Now press [Menu] once to turn on the un-shifted labels for the system-defined keys. The labels look like this:

```
| k0 | RECALL | CLR-END | CONT | ▒▒ |  STEP  | ALPHA | GRAPH | k9 |
```

The order of the labels on the screen matches the order of keys [f1] through [f8] on the keyboard.



Press [User] and notice that a U is now displayed in the lower right corner of the screen. Also notice that the labels have disappeared. Why? Because the user-defined keys are not defined (you are the user, and you have not yet defined them); thus, these keys have no labels. Press [System] again to return to system mode, and press [Menu] to bring back the labels for the system-defined keys.

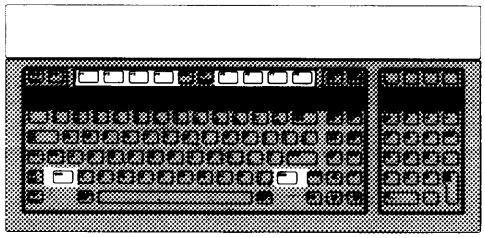
Now try out the [Menu] and [Shift]-[Menu] keys. Make sure you are in system mode and the labels are displayed. Press [Menu] repeatedly and notice how the un-shifted labels are turned on and off. Now press [Shift]-[Menu] once and watch the labels change to:

```
| k0 | RECALL | CLR-END | CONT | ▒▒ | ANYCHAR | DMP A | DMP G | k9 |
```

Press [Shift]-[Menu] repeatedly and see how the shifted labels are turned on and off. Press [Menu] to display the un-shifted labels again, then press [Menu] a second time to turn off the labels.

Note that displaying the menu inhibits the display of the type-ahead buffer, but does not inhibit the operation of the typeahead buffer itself.

System-Defined Keys



The system-defined keys are the definitions given to [f1] through [f8] while in system mode. Their actual labels are displayed on the screen when [Menu] or [Shift]-[Menu] is pressed.

RECALL — **RECALL** is used in the Debugger to recall the last command. See the Debugger chapter of the *Pascal Workstation System* manual for details.

CONT — **CONT** is used to exit the Debugger subsystem, or to continue normal operation after pressing [Break]. See the Debugger chapter of the *Pascal Workstation System* manual for details.

STEP — **STEP** is used in the Debugger for "stepping through" program execution, one line at a time. For more information, see the Debugger chapter in the *Pascal Workstation System* manual.

ANYCHAR — **ANYCHAR** is used to generate any ASCII or special character. To use it, first press **ANYCHAR** (Remember: **ANYCHAR** is a *shifted* system key, so you must press [Shift]-[f5] to get **ANYCHAR**). Then type any three digits from 000 through 255, representing the decimal equivalent of the character. The corresponding character will be generated; it may or may not be meaningful to the Pascal subsystem you're in. For instance, you can generate an asterisk (*) with **ANYCHAR** while in the Filer, but the Filer ignores it because it does not represent a valid Filer command.

To see **ANYCHAR** work, enter the Editor and press [I] to enter insert mode. Now press **ANYCHAR**, followed by the three digits 065. Notice the letter "A" is displayed on the screen, because 065 is the decimal equivalent (the ASCII character code) of the letter "A". Experiment with other numbers, if you like. When finished, press [ESC] to exit insert mode and clear the screen.

ALPHA
GRAPH

This key works for non-bitmap displays only.

The **ALPHA** and **GRAPH** keys allow you to turn the alpha and graphics display modes on and off. The **ALPHA** key turns *on* the *alphanumeric* display if you press it once, and turns *off* the *graphics* display if you press it a second time. The **GRAPH** key turns *on* the *graphics* display if you press it once, and turns *off* the *alphanumeric* display if you press it again.

To demonstrate these keys, exit the Editor, and enter the Filer. Press [v] to invoke the **Volumes** command. All on-line volumes will be listed on the screen.

Press **GRAPH** once to turn on the graphics display. (Unfortunately, there's nothing in the graphics display, so there is no visible change. Take our word for it: the graphics display is now on.) Press **GRAPH** again to turn off the alphanumeric display. The text on the screen disappears, including the labels for [f1] through [f8].

Now press **ALPHA** ([f6]) to turn on the alphanumeric display again. The text reappears. Press **ALPHA** once more to turn off the graphics display. Again, you'll have to take our word for it that the empty graphics display has now been turned off. Leave the **Volumes** display on the screen and go on to the next key demonstration.

DMP A

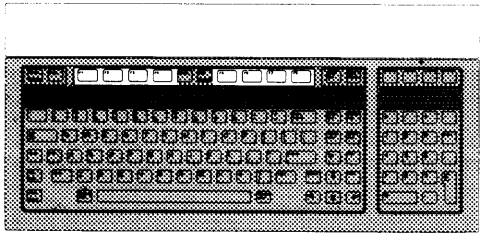
The **DMP A** key "dumps" whatever is in the alphanumeric display to your printer. Do not use **DMP A** if you have no printer. It may cause your system to "hang" for a while, as the computer searches for a non-existent printer.

If you have a printer, make sure it is properly connected and turned on. Also make sure the "on-line" light is on (if your printer has one). Now press **DMP A** ([Shift]-[f6]), and the **Volumes** display is printed on your printer.

DMP G

The **DMP G** key operates just like **DMP A**, except that it dumps whatever is in your graphics display to your printer. This key works properly only when you have a graphics printer. It is identical to **DMP A** for bitmap displays.

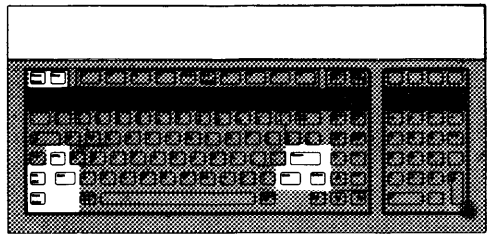
User-Defined Keys



The user-defined keys are the definitions given to [f1] through [f8] while in user mode. Their actual labels are displayed on the screen when [Menu] is pressed.

As their name suggests, user-defined keys must be defined by you, the user, or by an application program you buy. Instructions for defining keys are given in the *Pascal Procedure Library* manual.

System-Control Keys



Return

By far, the most frequently-used key is Return. Its function varies according to the context in which it is used. The most common uses include:

- Terminating your answer to a prompt.
- Aborting a prompt entered by mistake.
- In the Editor, moving the cursor to the beginning of the next line (a carriage return).

Note

The right-hand button on the mouse has the same functions as Return.

To demonstrate these uses, press F while in the Filer to get the **Filecopy** prompt:

```
Filecopy what file ?
```

Type:

```
ACCESS:EDITOR
```

Press [Return] to register your response. The computer now asks:

```
Filecopy to what?
```

Press [Return] with no answer to abort the prompt, since we really don't want to copy anything. Now exit the Filer and enter the Editor. Press [I] to enter insert mode. Insert mode allows you to type in text, just as you would on a typewriter. Type the following lines, and notice the [Return] key is just like RETURN on a typewriter:

```
THIS IS ONE LINE. [Return]
THIS IS ANOTHER LINE. [Return]
```

To exit insert mode without keeping the lines you just typed, press [ESC].

[Enter]

[Enter] has the same functions as [Return].

[Select]

[Select] makes your actions permanent. To see what this means, press [I] to put the Editor in insert mode. Type the following lines again:

```
THIS IS ONE LINE. [Return]
THIS IS ANOTHER LINE. [Return]
```

Now press [Back space] a few times and notice that every time the cursor backs up, it deletes one letter. The lines are not yet permanent. Now, retype the portion of the line you erased, then press [Select]. Now when you press [Back space], letters are not erased: they have become permanent. Leave this text on the screen for use in the upcoming demonstrations.

Note

The left-hand button on the mouse has the same function as [Select].

Shift-Select

Shift-Select (i.e., hold down Shift and press Select) nullifies any operation that has not yet been made permanent with the Select key. It also aborts certain prompts that you enter inadvertently, in a way similar to Return.

To see how Shift-Select nullifies an operation that is not yet permanent, hold down the space bar until the cursor is positioned at the end of the last line on the screen. Now press D to enter delete mode. Press Back space a few times to delete some characters at the end of the last line. Now press Shift-Select. Notice that the characters you deleted have returned, because Shift-Select nullified the delete operation. Understand, this would not be possible had you pressed Select first, for Select would have made the deletions permanent.

To see how Shift-Select can abort certain prompts, press the F key to invoke the **Find** command. **Find** is used to find any arbitrary sequence of characters you specify, but in this instance, let's assume you invoked the command by mistake. Notice the prompt line at the top of the screen:

```
Find[1]: L<target>=>
```

Now press Shift-Select and watch the prompt disappear. The **Find** operation has been canceled.

ESC generates an escape character, which behaves just like Shift-Select in most parts of the Pascal system.

CTRL

CTRL, pressed in conjunction with another key, generates a control code. To demonstrate this, press I to enter insert mode. Position the cursor below the last line on the screen and type:

```
THIS IS YET ANOTHER LINE.
```
Return

Now hold down CTRL and press C. Notice that insert mode is exited and the new line has become permanent. Thus CTRL-C has the same function as Select; in fact, Select generates the control code CTRL-C.

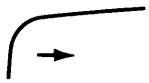
[Reset] allows you to boot another language or operating system without turning off your computer. Pressing [Reset] gives control to the Boot ROM after a few seconds. The self-test is rerun, and the computer begins searching for a system to load. It's as if you turned your computer off, then on again.

If the Debugger is loaded, [Reset] causes the computer to enter it.

Note

Press [f4] (**CONT**) to exit the Debugger.

The Debugger display is blank except for a small arrow in the upper-left corner of the screen:



If you would like to demonstrate this, press [Reset], but be prepared to re-boot the Pascal system according to the instructions in Chapter 2. If the Debugger is loaded, you press [Reset], then type the following line to cause the computer to reboot:

```
sb [Return]
```

`sb` stands for "system boot."

If the Debugger subsystem is loaded, [Break] suspends execution of whatever program is running. It, in effect, "pauses" the system.

[CTRL]-[Break] is the preferred way to enter the Debugger. See the Debugger chapter of the *Pascal Workstation System Manual* for details.

[Stop]

[Stop] immediately returns control to the Main Command Level, aborting any program that was running. It is a good way to exit an operation that you began by mistake, **but be careful not to press [Stop] while a disc read or write operation is in progress.**

If you rebooted your computer while trying out the [Reset] key, enter the Editor, press [I], and type the following lines again:

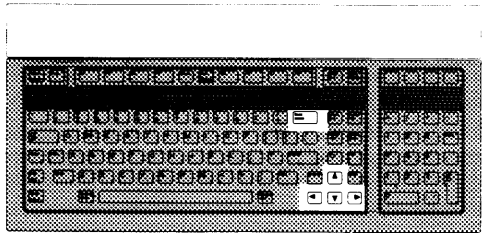
```
THIS IS ONE LINE. [Return]
THIS IS ANOTHER LINE. [Return]
THIS IS YET ANOTHER LINE. [Return]
```

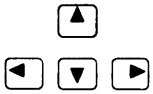
Press [Select] to make these lines permanent, then press [Stop]. Notice that we lied to you: the Editor is *not* immediately exited, but this is the only exception to the rule. Instead the following prompt appears on the screen:

```
Are you sure you want to STOP without updating?
     Type Yes  to STOP Without Update
     Type No   to Return To Editor
```

Since you have started a file in the Editor, the computer wants to confirm that you really do want to stop without updating, i.e., stop without saving this text on a disc. If you do not update, all of your work will be lost. Since we'll be using this text in subsequent demonstration, press [N]. The command line of the Editor now reappears on the screen.

Cursor-Control Keys





The arrow keys move the cursor in the indicated direction. Try them out on the text you've typed, moving the cursor up and down, left and right. Notice that the cursor will not move just anywhere on the screen, but only in the area of the text.

Back space

Back space moves the cursor backward one space. In insert mode, and when responding to prompts, it also erases characters as it backs up.

Use Back space to move backwards along one of the lines that you typed. Press I to enter insert mode and type a few characters. Then press Back space to erase all of the characters you typed. Press it once more, and the error message is generated at the top of the screen:

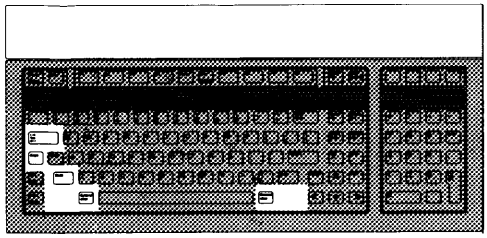
```
ERROR: Can't back up. <space> continues.
```

So you can't back up farther than you moved forward in insert mode. As indicated on the screen, press the space bar to return to insert mode, then press Select to exit.

Note

The mouse may be used as a cursor-control device. Simply move the mouse in the direction you would like the cursor to move.

Typewriter Keys



The keyboard has a full set of conventional typewriter keys, including a few special function keys described below.

[Caps]

[Caps] changes the case of all unshifted typewriter keys. When the Pascal system is booted, all letters are typed in upper case, with [Shift] providing the lower-case letters. Press [Caps], and all letters are typed in lower case, with [Shift] providing the upper case letters.

Press [I] to enter insert mode, and type a few characters, pressing [Shift] occasionally to provide the alternate case. Now press [Caps], and continue typing. Notice the case change. Press [ESC] to exit insert mode and clear what you typed.

[Shift]

[Shift] pressed in conjunction with another typewriter key produces the alternate case for that character; what is the alternate case depends on the setting of [Caps]. For non-typewriter keys, [Shift] provides access to the top-most function on the key.

[Tab]

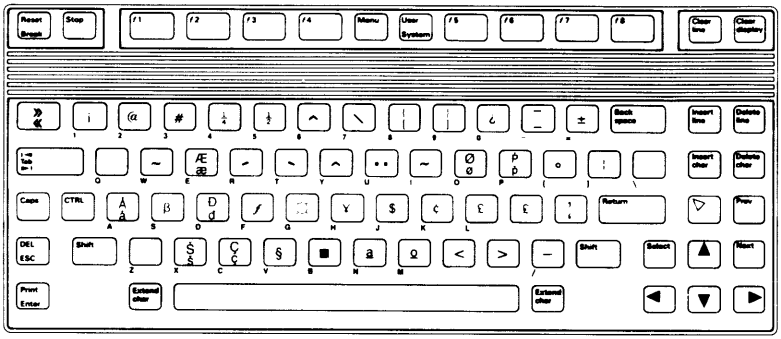
In the Editor, the [Tab] key moves the cursor ahead 8 spaces. Using the arrow keys, move the cursor to the beginning of one of the lines on the screen. Press [Tab] and watch the cursor jump forward.

Note

The [Tab] key is predefined to eight-space intervals; you cannot set it to a different value.

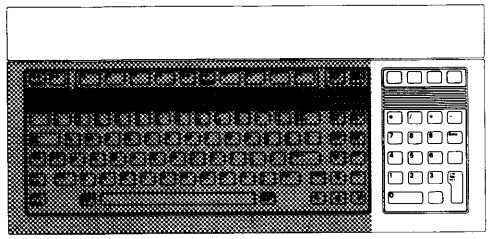
Extend char

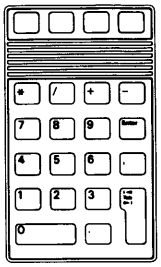
Extend char pressed in conjunction with another key accesses the Roman 8 character set. The complete Roman 8 character set "keyboard map" is provided below. Your computer may not be able to display all of these characters; when the key for an undisplayable character is pressed, an **hp** or regular alphabetic character will appear on the screen in its place.



Press I to enter insert mode. Hold down Extend char and press L. Notice that the Roman 8 character {pound sign} is displayed. Experiment with the other keys, if you like. When finished, press Shift-Select to abort what you've typed.

Numeric Keypad





The keyboard has a numeric keypad to permit fast entry of numeric data and easy execution of arithmetic operations.

* is provided for multiplication

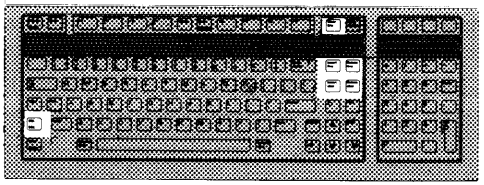
/ is provided for division

+ is provided for addition

- is provided for subtraction.

The Debugger subsystem can be used as a simple integer calculator. For details, consult the Debugger chapter in the *Pascal Workstation System Manual*

Editing Keys



The Pascal Editor can be operated with nothing more than typewriter keys. However, some special keyboard editing keys are implemented; use them if you find them more convenient.

[Insert line]

Pressing [Insert line] at the command level of the Editor is identical to pressing [I]—both put the Editor into insert mode.

[Insert char]

[Insert char] has the same function as [Insert line].

[Delete line]

Pressing [Delete line] at the command level of the Editor is identical to pressing [D]—both put the Editor into delete mode.

[Delete char]

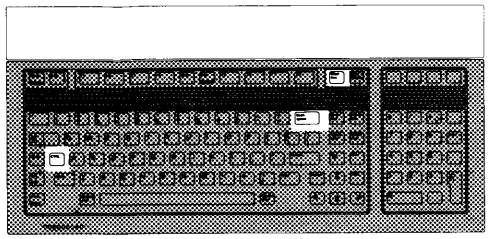
[Delete char] has the same function as [Delete line].

[Clear line]

[Clear line] erases the line containing the cursor while in insert mode in the Editor. Press [I] to enter insert mode, and type a few lines, pressing [Return] each time you want to start a new line. Press [Clear line] and notice that the line containing the cursor is removed. All but the first line you typed may be cleared this way. Press [Shift]-[Select] to exit insert mode.

[DEL] has the same function as [Clear line].

Type-ahead Buffer Keys



CTRL-Clear line

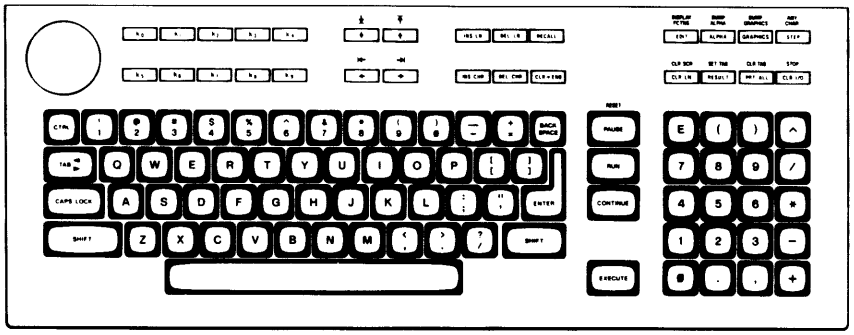
CTRL-Clear line removes *all* characters from the type-ahead buffer. See the "Type-ahead Buffer" section earlier in this chapter for a demonstration.

CTRL-Back space

CTRL-Back space deletes the *last* character entered into the type-ahead buffer. See the "Type-ahead Buffer" section earlier in this chapter for a demonstration.

The HP 98203B/C Keyboard

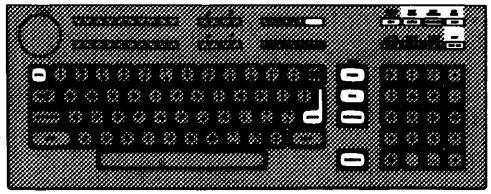
This section describes and demonstrates all of the keys Pascal defines on the Model 226/236 keyboards, the HP 98203B keyboard for the Model 216 and 220, and the HP 98203C keyboard for Series 300 computers.



Note

The "cursor" referred to in the following demonstrations is the underline character that appears on the screen. The cursor marks the position where the next typed character will be displayed.

System-Control Keys



[EDIT]

Pressing [EDIT] is the same as pressing [E]: it invokes the Editor if it is on-line. Since the [E] is in a more convenient position than [EDIT], [EDIT] is seldom used.

[ANY CHAR]
[STEP]

[ANY CHAR] is used to generate any ASCII or special character. To use it, first press [ANY CHAR]. Then type any three digits from 000 through 255, representing the decimal equivalent of the character. The corresponding character will be generated; it may or may not be meaningful to the Pascal subsystem you're in. For instance, you can generate an asterisk (*) with [ANY CHAR] while in the Filer, but the Filer ignores it because it does not represent a valid Filer command.

To see [ANY CHAR] work, press [I] to enter insert mode. Now press [ANY CHAR], followed by the three digits 065. Notice the letter "A" is displayed on the screen, because 065 is the decimal equivalent (the ASCII character code) of the letter "A". Experiment with other numbers, if you like. When finished, press [SHIFT]-[EXECUTE] to exit insert mode and clear the screen.

[ALPHA]
[GRAPHICS]

The [ALPHA] and [GRAPHICS] keys allow you to turn the alpha and graphics display modes on and off. The [ALPHA] key turns *on* the *alphanumeric* display if you press it once, and turns *off* the *graphics* display if you press it a second time. The [GRAPHICS] key turns *on* the *graphics* display if you press it once, and turns *off* the *alphanumeric* display if you press it again.

The [ALPHA] and [GRAPHICS] keys are not enabled on a 98203C keyboard except when used with a Model 217 computer.

To demonstrate these keys, exit the Editor, and enter the Filer. Press [V] to invoke the **Volumes** command. All on-line volumes will be listed on the screen.

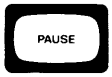
Press [GRAPHICS] once to turn on the graphics display. (Unfortunately, there's nothing in the graphics display, so there is no visible change. Take our word for it: the graphics display is now on.) Press [GRAPHICS] again to turn off the alphanumeric display. The text on the screen disappears.

Now press [ALPHA] to turn on the alphanumeric display again. The text reappears. Press [ALPHA] once more to turn off the graphics display. Again, you'll have to take our word for it that the empty graphics display has now been turned off. Leave the **Volumes** display on the screen and go on to the next key demonstration.

The [DUMP ALPHA] key "dumps" whatever is in the alphanumeric display to your printer. Do not use **DMP A** if you have no printer. It may cause your system to "hang" for a while, as the computer searches for a non-existent printer.

If you have a printer, make sure it is properly connected and turned on. Also make sure the "on-line" light is on (if your printer has one). Now press [DUMP ALPHA], and the **Volumes** display is printed on your printer.

For non-bitmap displays, the [DUMP GRAPHICS] key operates just like [DUMP ALPHA], except that it dumps whatever is in your graphics display to your printer. This key works properly only when you have a graphics printer.



If the Debugger subsystem is loaded, [PAUSE] suspends execution of whatever program is running. It, in effect, "pauses" the system.

[CTRL]-[PAUSE] is the preferred way to enter the Debugger. See the Debugger chapter of the *Pascal Workstation System Manual* for details.

[CONTINUE] is used to exit the Debugger subsystem, or to resume normal operation after pressing [PAUSE]. See the Debugger chapter of the *Pascal Workstation System Manual* for details.

[RUN] loads and runs the most recently compiled program. If no program has been compiled, [RUN] behaves just like the **eXecute** command in the Main Command Level—it prompts you for the name of the program file you want to run.

To see this, make sure the computer is at the Main Command Level and press [RUN]. Since we have not compiled a program, the following prompt will appear:

```
Execute what file ?
```

Press [ENTER] to abort the prompt. Now press [X] to invoke the **eXecute** command in the command line. The same prompt appears. Press [ENTER] again to abort the prompt.

[STEP] is used in the Debugger for "stepping through" program execution, one line at a time. For more information, see the Debugger chapter in the *Pascal Workstation System Manual.*

By far, the most frequently-used key is [ENTER]. Its function varies according to the context in which it is used. The most common uses include:

- Terminating your answer to a prompt.
- Aborting a prompt entered by mistake.
- In the Editor, moving the cursor to the beginning of the next line (a carriage return).

To demonstrate these uses, press [F] while in the Filer to get the **Filecopy** prompt:

```
Filecopy what file ?
```

Type:

```
ACCESS:EDITOR
```

Press [ENTER] to register your response. The computer now asks:

```
Filecopy to what?
```

Press [ENTER] with no answer to abort the prompt, since we really don't want to copy anything. Now exit the Filer and enter the Editor. Press [I] to enter insert mode in the Editor. Insert mode allows you to type in text, just as you would on a typewriter. Type the following lines, and notice the [ENTER] key is just like RETURN on a typewriter:

```
THIS IS ONE LINE. [ENTER]
THIS IS ANOTHER LINE. [ENTER]
```

To exit insert mode without keeping the lines you just typed, press [SHIFT]-[EXECUTE].

[EXECUTE] makes your actions permanent. To see what this means, press [I] to put the Editor in insert mode. Type the following lines again:

```
THIS IS ONE LINE. [ENTER]
THIS IS ANOTHER LINE. [ENTER]
```

Now press [BACK SPACE] a few times and notice that every time the cursor backs up, it deletes one letter. The lines are not yet permanent. Now, retype the portion of the line you erased, then press [EXECUTE]. Now when you press [BACK SPACE], letters are not erased; they have become permanent. Leave this text on the screen for use in the upcoming demonstration.

[SHIFT]-[EXECUTE] (i.e., hold down [SHIFT] and press [EXECUTE]) nullifies any operation that has not yet been made permanent with the [EXECUTE] key. It also aborts certain prompts that you enter inadvertently, in a way similar to [ENTER].

To see how [SHIFT]-[EXECUTE] nullifies an operation that is not yet permanent, hold down the space bar until the cursor is positioned at the end of the last line on the screen. Now press [D] to enter delete mode. Press [BACK SPACE] a few times to delete some characters at the end of the last line. Now press [SHIFT]-[EXECUTE]. Notice that the characters you deleted have returned, because [SHIFT]-[EXECUTE] nullified the delete operation. Understand, this would not be possible had you pressed [EXECUTE] first, for [EXECUTE] would have made the deletions permanent.

To see how [SHIFT]-[EXECUTE] can abort certain prompts, press the [F] key to invoke the **Find** command. **Find** is used to find any arbitrary sequence of characters you specify, but in this instance, let's assume you invoked the command by mistake. Notice the prompt line at the top of the screen:

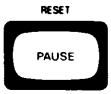
```
Find[1]: L<target>=>
```

Now press [SHIFT]-[EXECUTE] and watch the prompt disappear. The Find operation has been canceled.

[CTRL], pressed in conjunction with another key, generates a control code. To demonstrate this, press [I] to enter insert mode. Position the cursor below the last line on the screen and type:

```
THIS IS YET ANOTHER LINE. [ENTER]
```

Now hold down [CTRL] and press [C]. Notice that insert mode is exited and the new line has become permanent. Thus [CTRL]-[C] has the same function as [EXECUTE]; in fact, [EXECUTE] generates the control code CTRL-C.



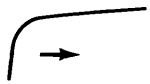
[RESET] ([SHIFT]-[PAUSE]) allows you to boot another language/operating system without turning off your computer. Pressing [RESET] gives control to the Boot ROM after a few seconds, which reruns the self-test and then begins searching for a system to load. It's as if you turned your computer off, then on again.

If the Debugger is loaded, [RESET] causes the computer to enter it; although [CTRL]-[PAUSE] is the preferred way to enter the debugger.

Note

Press [CONTINUE]) to exit the Debugger.

The Debugger display is blank except for a small arrow in the upper-left corner of the screen:



If you would like to demonstrate this, press [RESET], but be prepared to re-boot the Pascal system according to the instructions in Chapter 2. If the Debugger is loaded, you must press [RESET], then type the following line to cause the computer to reboot:

```
sb [ENTER]
```

`sb` stands for "system boot."

STOP
CLR I/O

[STOP] immediately returns control to the Main Command Level, aborting any program that was running. It is a good way to exit an operation that you began by mistake, but be careful not to press [STOP] while a disc read or write operation is in progress.

If you rebooted your computer while trying out the [RESET] key, enter the Editor, press [I], and type the following lines again:

```
THIS IS ONE LINE. [ENTER]
THIS IS ANOTHER LINE. [ENTER]
THIS IS YET ANOTHER LINE. [ENTER]
```

Press [EXECUTE] to make these lines permanent, then press [STOP]. Notice that we lied to you: the Editor is *not* immediately exited, but this is the only exception to the rule. Instead the following prompt appears on the screen:

```
Are you sure you want to STOP without updating?
     Type Yes  to STOP Without Update
     Type No   to Return To Editor
```

Since you have started a file in the Editor, the computer wants to confirm that you really do want to stop without updating, i.e., stop without saving this text on a disc. If you do not update, all of your work will be lost. Since we'll be using this text in subsequent demonstrations, press [N]. The command line of the Editor now reappears on the screen.

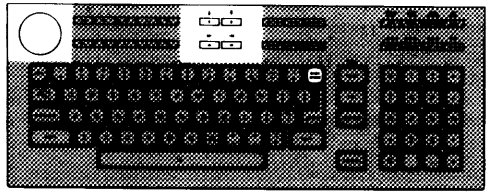
CLR I/O

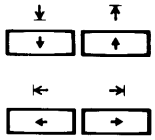
[CLR I/O] has the same function as [STOP].

RECALL

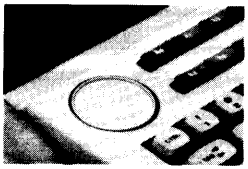
[RECALL] is used in the Debugger to recall the last command. See the Debugger chapter of the *Pascal Workstation System Manual* for details.

Cursor-Control Keys





The arrow keys move the cursor in the indicated direction. Try them out on the text you've typed, moving the cursor up and down, left and right. Notice that the cursor will not move anywhere on the screen, but only in the area of the text.



The knob is used for rapid movement through text. Turn the knob clockwise and watch the cursor move forward along your lines of text. Turn the knob counter-clockwise, and the cursor moves backward through the text. Hold down the [SHIFT] key and rotate the knob clockwise and the cursor moves rapidly down through lines of text. Hold down [SHIFT] and rotate the knob counter-clockwise and the cursor moves rapidly upward through lines of text.

Note that to use the knob on the HP 98203C keyboard will require installing the `HPHIL` and `MOUSE` drivers.

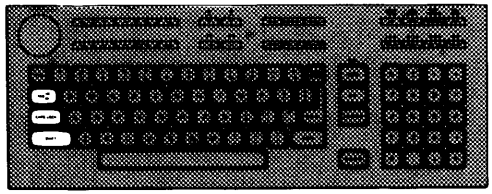
[BACK SPACE] moves the cursor backward one space. In insert mode, and when responding to prompts, it also erases characters as it backs up.

Use [BACK SPACE] to move backwards along one of the lines that you typed. Press [I] to enter insert mode and type a few characters. Then press [BACK SPACE] to erase all of the characters you typed. Press it once more, and the error message is generated at the top of the screen:

```
ERROR: Can't back up. <space> continues.
```

So you can't back up farther than you moved forward in insert mode. As indicated on the screen, press the space bar to return to insert mode, then press [EXECUTE] to exit.

Typewriter Keys



The keyboard has a full set of conventional typewriter keys, included a few special function keys described below.

[CAPS LOCK] changes the case of all unshifted typewriter keys. When the Pascal system is booted, all letters are typed in upper case, with [SHIFT] providing the lower-case letters. Press [CAPS LOCK], and all letters are typed in lower case, with [SHIFT] providing the upper case letters.

Press [I] to enter insert mode, and type a few characters, pressing [SHIFT] occasionally to provide the alternate case. Now press [CAPS LOCK], and continue typing. Notice the case change. Press [SHIFT]-[EXECUTE] to exit insert mode and clear what you typed.

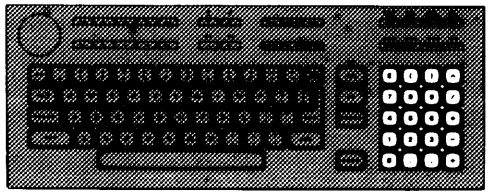
[SHIFT] pressed in conjunction with another typewriter key produces the alternate case for that character; what is the alternate case depends on the setting of [CAPS LOCK]. For non-typewriter keys, [SHIFT] provides access to the top-most function on the key.

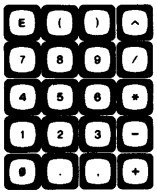
In the Editor, the [TAB] key moves the cursor ahead 8 spaces. Using the arrow keys, move the cursor to the beginning of one of the lines on the screen. Press [TAB] and watch the cursor jump forward.

Note

The [TAB] key is predefined to eight-space intervals; you cannot set it to a different value.

Numeric Keypad





The keyboard has a numeric keypad to permit fast entry of numeric data and easy execution of arithmetic operations.

* is provided for multiplication

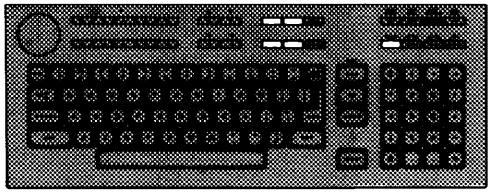
/ is provided for division

+ is provided for addition

- is provided for subtraction.

The Pascal Debugger subsystem can be used as a simple integer calculator. For details, consult the Debugger chapter in the *Pascal Workstation System* manual.

Editing Keys



The Pascal Editor can be completely operated with only the typewriter keys. However, some special keyboard editing keys are implemented; use them if you find them more convenient.

INS LN

Pressing [INS LN] at the command level of the Editor is identical to pressing [I]—both put the Editor into insert mode.

INS CHR

[INS CHR] has the same function as [INS LN].

DEL LN

Pressing [DEL LN] at the command level of the Editor is identical to pressing [D]—both put the Editor into Delete mode.

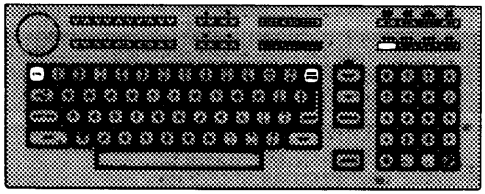
DEL CHR

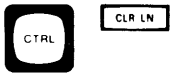
[DEL CHR] has the same function as [DEL LN].

CLR LN

[CLR LN] erases the line containing the cursor while in insert mode in the Editor. Press [I] to enter insert mode, and type a few lines of characters, pressing [ENTER] each time you want to start a new line. Press [CLR LN] and notice that the line containing the cursor is removed. All but the first line typed may be cleared this way. Press [SHIFT]-[EXECUTE] to exit insert mode.

Type-ahead Buffer Keys



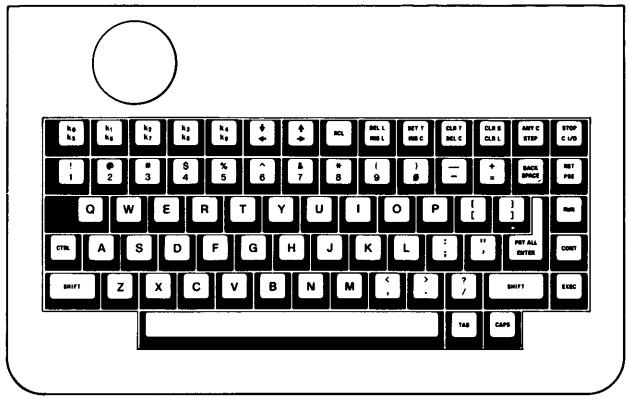


CTRL-CLR LN removes *all* characters from the type-ahead buffer. See the Type-ahead buffer section earlier in this chapter for a demonstration.

CTRL-BACK SPACE deletes the *last* character entered into the type-ahead buffer. See the Type-ahead buffer section earlier in this chapter for a demonstration.

The HP 98203A Keyboard

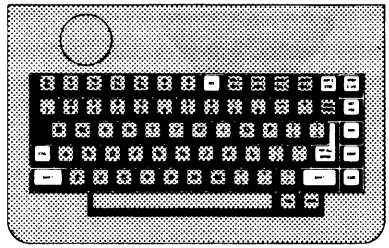
This section describes and demonstrates all of the keys Pascal defines on the HP 98203A keyboard for the Model 216 and 220.



Note

The "cursor" referred to in the following demonstrations is the underline character that appears on the screen. The cursor marks the position where the next typed character will be displayed.

System-Control Keys



[ANY C] is used to generate any ASCII or special character. To use it, first press [ANY C]. Then type any three digits from 000 through 255, representing the decimal equivalent of the character. The character will be generated; it may or may not be meaningful to the Pascal subsystem you're in. For instance, you can generate an asterisk (*) with [ANY C] while in the Filer, but the Filer ignores it because it does not represent a valid Filer command.

To see [ANY C] work, press [I] to enter the Editor's insert mode. Now press [ANY C], followed by the three digits **065**. Notice the letter "A" is displayed on the screen, because 065 is the decimal equivalent (called the ASCII character code) of the letter "A". Experiment with other numbers, if you like. When finished, press [SHIFT]-[EXEC] to exit insert mode and clear the screen.

If the Debugger subsystem is loaded, [PSE] suspends execution of whatever program is running. It, in effect, "pauses" the system.

[CTRL]-[PSE] is the preferred way to enter the Debugger. See the Debugger chapter of the *Pascal Workstation System Manual* for details.

[CONT] is used to exit the Debugger subsystem, or to resume normal operation after pressing [PSE]. See the Debugger chapter of the *Pascal Workstation System Manual* for details.

RUN loads and runs the most recently compiled program. If no program has been compiled, RUN behaves just like the **eXecute** command in the Main Command Level—it prompts you for the name of the program file you want to run.

To see this, make sure the computer is at the Main Command Level and press RUN. Since we have not compiled a program, the following prompt will appear:

```
Execute what file ?
```

Press ENTER to abort the prompt. Now press X to invoke the **eXecute** command in the command line. The same prompt appears. Press ENTER again to abort the prompt.

STEP is used in the Debugger for "stepping through" program execution, one line at a time. For more information, see the Debugger chapter in the *Pascal Workstation System Manual.*

By far, the most frequently-used key is ENTER. Its function varies according to the context in which it is used. The most common uses include:

- Terminating your answer to a prompt.
- Aborting a prompt entered by mistake.
- In the Editor, moving the cursor to the beginning of the next line (a carriage return).

To demonstrate these uses, press F while in the Filer to get the Filecopy prompt:

```
Filecopy what file ?
```

Type:

```
ACCESS:EDITOR
```

Press ENTER to register your response. The computer now asks:

```
Filecopy to what?
```

Press [ENTER] with no answer to abort the prompt, since we really don't want to copy anything. Now exit the Filer and enter the Editor. Press [I] to enter insert mode in the Editor. Insert mode allows you to type in text, just as you would on a typewriter. Type the following lines, and notice the [ENTER] key is just like RETURN on a typewriter:

```
THIS IS ONE LINE. [ENTER]
THIS IS ANOTHER LINE. [ENTER]
```

To exit insert mode without keeping the lines you just typed, press [SHIFT]-[EXEC].

[EXEC] is used to make your actions permanent. To see what this means, press [I] to put the Editor in insert mode. Type the following lines again:

```
THIS IS ONE LINE. [ENTER]
THIS IS ANOTHER LINE. [ENTER]
```

Now press [BACK SPACE] a few times and notice that every time the cursor backs up, it deletes one letter. The lines are not yet permanent. Now, retype the portion of the line you erased, then press [EXEC]. Now when you press [BACK SPACE], letters are not erased; they have become permanent. Leave this text on the screen for use in the upcoming demonstration.

[SHIFT]-[EXEC] (i.e., hold down [SHIFT] and press [EXEC]) nullifies any operation that has not yet been made permanent with the [EXEC] key. It also aborts certain prompts that you enter inadvertently, in a way similar to [ENTER].

To see how [SHIFT]-[EXEC] nullifies an operation that is not yet permanent, hold down the space bar until the cursor is positioned at the end of the last line on the screen. Now press [D] to enter Delete mode. Press [BACK SPACE] a few times to delete characters at the end of the last line. Now press [SHIFT]-[EXEC]. Notice that the characters you deleted have returned, because [SHIFT] [EXEC] nullified the delete operation. Understand, this would not be possible had you pressed [EXEC] first, for [EXEC] would have made the deletions permanent.

To see how [SHIFT]-[EXEC] can abort certain prompts, press the [F] key to invoke the **Find** command. **Find** is used to find any arbitrary sequence of characters you specify, but in this instance, let's assume you invoked the command by mistake. Notice the prompt line at the top of the screen:

```
Find[1]: L<target>=>
```

Now press [SHIFT]-[EXEC] and watch the prompt disappear. The Find operation has been canceled.

[CTRL], pressed in conjunction with another key, generates a control code. To demonstrate this, press [I] to enter insert mode. Position the cursor below the last line on the screen and type:

```
THIS IS YET ANOTHER LINE. [ENTER]
```

Now hold down [CTRL] and press [C]. Notice that insert mode is exited and the new line has become permanent. Thus [CTRL]-[C] has the same function as [EXEC]; in fact, [EXEC] generates the control code CTRL-C.

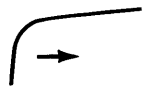
[RST] allows you to boot another language/operating system without turning off your computer. Pressing [RST] gives control to the Boot ROM after a few seconds, which reruns the self-test and then begins searching for a system to load. It's as if you turned your computer off, then on again.

If the Debugger is loaded, [RST] causes the computer to enter it; however, [CTRL]-[PSE] is the preferred way to enter the debugger.

Note

Press (**CONT**) to exit the Debugger.

The Debugger display is blank except for a small arrow in the upper-left corner of the screen:



If you would like to demonstrate this, press [RST], but be prepared to re-boot the Pascal system according to the instructions in Chapter 2. If the Debugger is loaded, you must press [RST], then type the following line to cause the computer to reboot:

```
sb [ENTER]
```

`sb` stands for "system boot."

[STOP] immediately returns control to the Main Command Level, aborting any program that was running. It is a good way to exit an operation that you began by mistake, but be careful not to press [STOP] while a disc read or write operation is in progress.

If you rebooted your computer while trying out the [RST] key, enter the Editor, press [I], and type the following lines again:

```
THIS IS ONE LINE. [ENTER]
THIS IS ANOTHER LINE. [ENTER]
THIS IS YET ANOTHER LINE. [ENTER]
```

Press [EXEC] to make these lines permanent, then press [STOP]. Notice that we lied to you: the Editor is *not* immediately exited, but this is the only exception to the rule. Instead the following prompt appears on the screen:

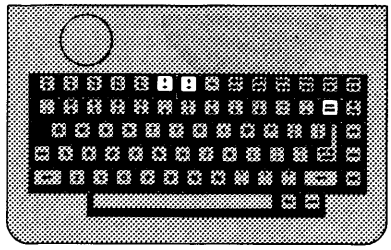
```
Are you sure you want to STOP without updating?
     Type Yes  to STOP Without Update
     Type No   to Return To Editor
```

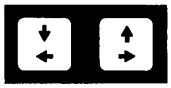
Since you have started a file in the Editor, the computer wants to confirm that you really do want to stop without updating, i.e., stop without saving this text on a disc. If you do not update, all of your work will be lost. Since we'll be using this text in subsequent demonstrations, press [N]. The command line of the Editor now reappears on the screen.

[C I/O] has the same function as [STOP].

[RCL] is used in the Debugger to recall the last command. See the Debugger chapter of the *Pascal Workstation System Manual* for details.

Cursor-Control Keys





The arrow keys move the cursor in the indicated direction. Try them out on the text you've typed, moving the cursor up and down, left and right. Notice that the cursor will not move anywhere on the screen, but only in the area of the text.

The knob is used for rapid movement through text. Turn the knob clockwise and watch the cursor move forward along your lines of text. Turn the knob counter-clockwise, and the cursor moves backward through the text. Hold down the [SHIFT] key and rotate the knob clockwise and the cursor moves rapidly down through lines of text. Hold down [SHIFT] and rotate the knob counter-clockwise and the cursor moves rapidly upward through lines of text.

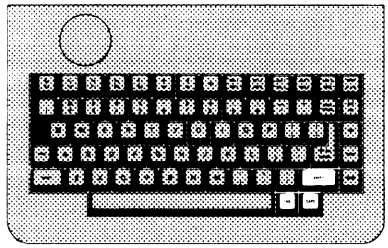
[BACK SPACE] moves the cursor backward one space. In insert mode, and when responding to prompts, it also erases characters as it backs up.

Use [BACK SPACE] to move backwards along one of the lines that you typed. Press [I] to enter insert mode and type a few characters. Then press [BACK SPACE] to erase all of the characters you typed. Press it once more, and the error message is generated at the top of the screen:

```
ERROR: Can't back up. <space> continues.
```

So you can't back up farther than you moved forward in insert mode. As indicated on the screen, press the space bar to return to insert mode, then press [EXEC] to exit.

Typewriter Keys



The keyboard has a full set of conventional typewriter keys, included a few special function keys described below.

[CAPS] changes the case of all unshifted typewriter keys. When the Pascal system is booted, all letters are typed in upper case, with [SHIFT] providing the lower-case letters. Press [CAPS], and all letters are typed in lower case, with [SHIFT] providing the upper case letters.

Press [I] to enter insert mode, and type a few characters, pressing [SHIFT] occasionally to provide the alternate case. Now press [CAPS], and continue typing. Notice the case change. Press [SHIFT]-[EXEC] to exit insert mode and clear what you typed.

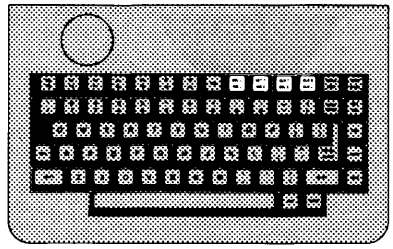
[SHIFT] pressed in conjunction with another typewriter key produces the alternate case for that character; what is the alternate case depends on the setting of [CAPS]. For non-typewriter keys, [SHIFT] provides access to the top-most function on the key.

In the Editor, the [TAB] key moves the cursor ahead 8 spaces. Using the arrow keys, move the cursor to the beginning of one of the lines on the screen. Press [TAB] and watch the cursor jump forward.

Note

The [TAB] key is predefined to eight-space intervals; you cannot set it to a different value.

Editing Keys



The Pascal Editor can be completely operated with nothing more than typewriter keys. However, some special keyboard editing keys are implemented; use them if you find them more convenient.

Pressing [INS L] at the command level of the Editor is identical to pressing [I]—both put the Editor into insert mode.

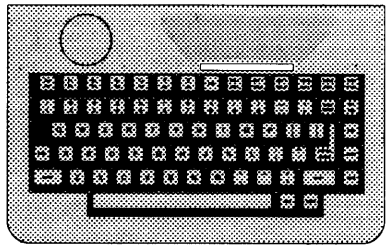
[INS C] has the same function as [INS L].

Pressing [DEL L] at the command level of the Editor is identical to pressing [D]—both put the Editor into Delete mode.

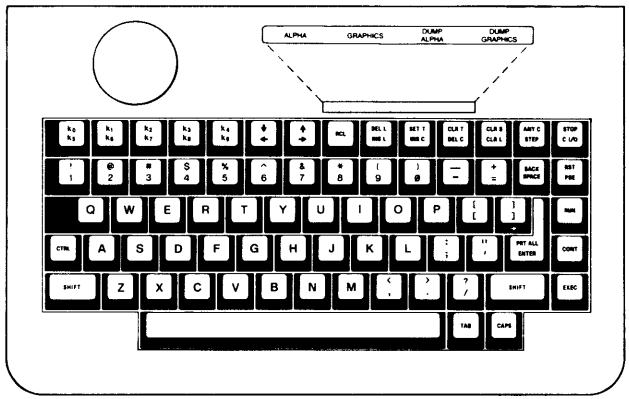
[DEL C] has the same function as [DEL L].

[CLR L] erases the line containing the cursor while in insert mode in the Editor. Press [I] to enter insert mode, and type a few lines of characters, pressing [ENTER] each time you want to start a new line. Press [CLR L] and notice that the line containing the cursor is removed. All but the first line typed may be cleared this way. Press [SHIFT]-[EXEC] to exit insert mode.

Keyboard Label Keys



The Pascal Keyboard Label was shipped in your computer's miscellaneous kit. Affix the label to the keyboard as shown below.



Affixing the Keyboard Label

To invoke one of the functions printed on the label, hold down [SHIFT] and press the key directly beneath the desired function. For example, to dump graphics, hold down [SHIFT] and press [DEL C].

The keyboard label keys are defined as follows:

Alpha
Graphics

The **Alpha** (shifted [RCL]) and **Graphics** (shifted [INS L]) keys allow you to turn the alpha and graphics display modes on and off. The **Alpha** key turns *on* the *alphanumeric* display if you press it once, and turns *off* the *graphics* display if you press it a second time. The **Graphics** key turns *on* the *graphics* display if you press it once, and turns *off* the *alphanumeric* display if you press it again.

To demonstrate these keys, enter the Filer and press **V** to invoke the **Vols** command. All on-line volumes will be listed on the screen. Press **GRAPHICS** (shifted [RCL]) once to turn on the graphics display. (Unfortunately, there's nothing in the graphics display, so there is no visible change. Take our word for it: the graphics display is now on.) Press **GRAPHICS** again to turn off the alphanumeric display. The text on the screen disappears.

Now press **ALPHA** (shifted [INS L]) to turn on the alphanumeric display again. The text reappears. Press **ALPHA** once more to turn off the graphics display. Again, you'll have to take our word for it that the empty graphics display has now been turned off. Leave the **Vols** display on the screen and go on to the next key demonstration.

DUMP ALPHA

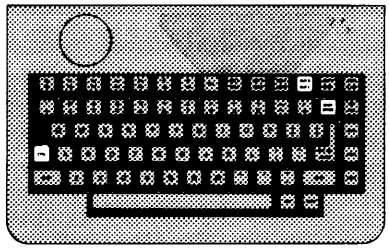
The **DUMP ALPHA** key (shifted [INS C]) "dumps" whatever is in the alphanumeric display to your printer. Do not use **DUMP ALPHA** if you have no printer; it may cause your system to "hang" for a while, as the computer searches for a non-existent printer.

If you do have a printer, make sure it is properly connected and turned on. Also make sure the "on-line" light is on (if your printer has one). Now press **DUMP ALPHA**, and the **Vols** display is printed on your printer.

DUMP GRAPHICS

The **DUMP GRAPHICS** key (shifted [DEL C]) operates just like **DUMP ALPHA**, except that it dumps whatever is in the graphics display to your printer. This key works properly only if you have a graphics printer.

Type-ahead Buffer Keys



CTRL-CLR L removes *all* characters from the type-ahead buffer. See the Type-ahead buffer section earlier in this chapter for a demonstration.

CTRL-BACK SPACE deletes the *last* character entered into the type-ahead buffer. See the Type-ahead buffer section earlier in this chapter for a demonstration.

The Main Command Level 5

The Main Command Level is the highest level of the Pascal Workstation—your home base and port-of-entry into the rest of the system. Using an analogy to the Pascal programming language, the Main Command Level is like the main program that calls all of the subordinate procedures and functions. It really does nothing itself, but summons up the appropriate subprogram to carry out the requested task.

The command line of the Main Command Level looks like this:

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

Press [?] and the rest of the commands appear:

```
Command: Assembler Debugger Memvol Newsysvol Permanent Stream User What ?
```

We'll now examine some of the most important commands.

Primary Main Command Level Commands

Editor

The Editor subsystem is invoked by pressing [E], and is stored on the ACCESS: volume under the file name EDITOR.

The Editor is used to create and edit programs and text. Specific Editor commands are discussed in Chapter 6.

Filer

The Filer subsystem is invoked by pressing [F], and is stored on the ACCESS: volume under the file name FILER.

The Filer is used to manage interactions between the computer and its mass storage devices (e.g., disc drives, bubble memory, etc.). Specific Filer commands are discussed in Chapter 7.

Compiler

The Compiler subsystem is invoked by pressing [C], and is stored on the CMP: volume under the file name COMPILER.

The Compiler is used to compile Pascal program text into loadable Series 200/300 object code. Specific operational details are provided in Chapter 8.

eXecute

The **eXecute** command is invoked by pressing [X]. It is used to run programs. The **eXecute** command is "built in" to the Main Command Level; no special volume must be on-line to use it.

In Chapter 3, you used **eXecute** to load and run the media initialization program stored in the file MEDIAINIT.CODE on the ACCESS: volume. Let's take a closer look at the behavior of the **eXecute** command.

Press [X]. The following prompt now appears:

```
Execute what file ?
```

The computer is asking for the name of the file containing the program you want to run. Remember that to completely specify a file, you must provide not only the *name* of the file, but its *location* (unit number or volume ID) as well. (Don't forget that file names on hierarchical systems such as SRM or HFS include any directory names leading to the file.) For example, if the program you want to run is in a file named NONSENSE on the ABSENT: volume, you would type:

```
ABSENT:NONSENSE [Return]
```

Go ahead and try this. Note that a : must always separate the volume ID (or unit number) from the file name.

The computer now responds with an error message:

```
Loading 'ABSENT:NONSENSE.CODE'
cannot open 'ABSENT:NONSENSE.CODE'
logical volume not found
```

The computer searched for the ABSENT: volume but could not find it. If you ever get this message while trying to run a program, check that you've spelled the volume ID correctly, and make sure the required volume is on-line. In this example, the latter problem applies, for there is no ABSENT: volume.

Notice that the computer appended the suffix, .CODE, to the file name. This is standard procedure. Because the Compiler creates files with names ending in .CODE when it compiles program text into executable code, the **eXecute** command assumes that most programs you want to run will be stored in a .CODE file. So you don't have to type the .CODE suffix every time; the computer will save you the trouble and append it automatically. However, what if you don't *want* to execute a .CODE file?

As an example, the Filer subsystem is nothing more than a program, but it is stored in a file named FILER, not in a file named FILER.CODE. Suppose you wanted to use **eXecute** to run the Filer, instead of invoking it with [F]. How do you keep the **eXecute** command from appending `.CODE` to the file name?

Insert the ACCESS: volume (ACCESS: contains the Filer) into one of your disc drives. Press [X] again and answer the prompt with:

```
ACCESS:FILER [Return]
```

You get the message:

```
Loading 'ACCESS:FILER.CODE'
cannot open 'ACCESS:FILER.CODE'
file not found
```

Of course it couldn't find FILER.CODE; there is no such file name on the ACCESS: volume. What you want is the file named FILER. To prevent the computer from appending `.CODE` to the file name, you must type a period (.) after the file name. The period is not interpreted as part of the file name, but only serves to suppress the suffix. Type:

```
ACCESS:FILER. [Return]
```

The Filer is loaded and run. Press [Q] to quit the Filer and return to the Main Command Level. Press any other key to clear the screen. Incidentally, you should watch for that last error message (`file not found`). If you run across it again, make sure you have spelled the file name correctly, that the file really does exist on the volume, and that the .CODE suffix is part of the actual file name.

What

To **eXecute** the Filer in the last example, you specified its volume ID and file name. Without this information, the computer has no idea where to find the Filer. How, then, does computer know where to find the Filer when you press the [F] key?

The computer keeps track of the volume IDs and file names of the Filer and all other subsystems in a table. So when you press [F], for example, the computer looks up the Filer in this table, finds its file name and volume ID, and tries to retrieve it. If the information in the table is correct, it finds and loads the Filer. If it's not, it displays an error message.

You can look at and modify this table by invoking the **What** command. Press [W] now to see the table. It will look something like this, although each particular entry depends on how you have your system arranged:

```
Assembler Compiler Editor Filer Librarian
liBrary System volume Default volume Quit

ASSEMBLER:   ASM:ASSEMBLER
COMPILER:    CMP:COMPILER
EDITOR:      ACCESS:EDITOR
FILER:       ACCESS:FILER
LIBRARIAN:   ACCESS:LIBRARIAN
LIBRARY:     SYSVOL:LIBRARY

* System volume:  SYSVOL:
: Default volume: BOOT:
```

The two lines at the top of the screen are command lines; there is one command for each entry in the table. Each command allows you to change the volume ID and file name for the corresponding entry.

Press [A] to see how this works, but don't type anything. The ASSEMBLER entry goes blank, and the computer waits for you to type in the new volume ID and file name. Press [Return], and the original entry is restored.

Each entry in the table gives the volume name and the file name where the computer *expects* to find that subsystem. If your Pascal system is on flexible discs, these volume IDs correspond to your disc labels—CMP: for the disc containing the Compiler; ACCESS: for the disc containing the Editor, Filer and Librarian; etc. If your Pascal workstation is part of an SRM system, these volume IDs read SYSTEM*nn*, where *nn* is the node address of your workstation's SRM interface.

The two lines at the bottom give the settings for the default and system volumes.

The *default volume* is the volume ID the computer will "assume" if you leave off the volume ID when specifying a file. For example, remember when running the media initialization program in Chapter 3, you typed the following complete file specification when prompted with **Execute what file ?**:

```
ACCESS:MEDIAINIT
```

It is tiresome to type the volume ID if you're accessing the same volume all the time, so set the default volume to the volume ID of the frequently-accessed volume. To set the default volume to ACCESS:, press [D] and type:

```
ACCESS: [Return]
```

Now, to see how the computer behaves when no volume is specified, press [Q] to return to the Main Command Level and press [X] to invoke the **eXecute** command. Now type only the file name (MEDIAINIT) in response to the prompt:

```
MEDIAINIT [Return]
```

The computer assumes MEDIAINIT must be on the default volume, and constructs the complete file specification using ACCESS: as the volume ID. The media initialization program is loaded and run. Press [Return] to abort it, and press [W] to return to the **What** command.

The *system volume* is used by the Pascal system to store temporary files, special files, the date that you last entered, and other information. Since the system volume is important to the operation of the Pascal Workstation, it should be on-line at all times. Setting the system volume is described in the section for the **Newsysvol** command later in this chapter, and in Part III of this book.

Both the system volume and the default volume have a special character associated with them. The system volume has an asterisk (*); the default volume has a colon (:). Each character is just a shorthand notation for that volume, and can be used in place of its volume ID. You'll see the convenience of this feature in Chapter 7.

Before you leave the **What** command, change the volume ID of the Filer to see what happens when the information in the table becomes corrupted. But before you do this, write down the correct entry so you can restore it when you're finished.

Press [F] to change the FILER entry. Type:

```
BADVOL:FILER. [Return]
```

Note that you must still append the period to keep the suffix .CODE from being automatically appended. Now press [Q] to exit the **What** command, and press [F] to load the Filer.

What happened? The computer looked in the table, pulled out the entry for the Filer, looked for the BADVOL: volume and couldn't find it. It reports:

```
Loading 'BADVOL:FILER'
cannot open 'BADVOL:FILER'
logical volume not found
```

Press [W], and change the Filer entry back to its original value. Press [Q] to return to the Main Command Level, and confirm that pressing [F] successfully loads the Filer again. If for some reason it doesn't, and you can't change the Filer entry back to its original value, simply reboot the Pascal system. The booting process sets all of the initial values in the table to their proper values.

Run

The **Run** command is invoked by pressing either [R] or **Run**. **Run** works just like the **eXecute** command, unless there is a *workfile* present. workfiles will be discussed in detail later, but for now think of it as a default file: one the computer "assumes" you want to access.

Suppose you just finished typing in a Pascal program and saved it as your workfile. To run this program, you can just press [R], and the program will be compiled into executable code and then executed. This single step is easier and faster than saving your program text in a file, then compiling the text file and generating a code file, and finally using **eXecute** to run the code file.

If your workfile already contains compiled code, the compilation step is skipped and the workfile is immediately run. **Run** will be demonstrated in Part III.

User

The **User** command is invoked by pressing [U]. It simply re-runs the last program you ran. Programs, in this case, include the Pascal subsystems, such as the Filer and Editor.

You used the **User** command in Chapter 3 to re-start the media initialization program when you were ready to initialize another disc. It saved you from going back to the **eXecute** command each time and specifying which program you wanted to run. Since the program was just run, there was no need to re-load and re-eXecute it.

Permanent

Permanent is invoked by pressing [P]. It permanently loads a program into memory.

Up to now, every time you invoked the Filer or the Editor, it took several seconds to load the subsystem from the disc into memory. There could be only one subsystem in memory at a time, and every time you invoked a new subsystem, the old one was lost. It would be much better to have your most frequently-used subsystems *permanently* loaded into memory. The **Permanent** command does just that.

To understand the convenience of permanent-loading, try out this demonstration. With your ACCESS: volume on-line, press [E] to invoke the Editor. Note the time delay as the Editor is loaded from the volume into memory. Press [Return] and then [Q] to quit the Editor, then press [F] to invoke the Filer. The Filer is loaded into memory, replacing the Editor. Quit the Filer, and invoke the Editor again. You'll probably agree that the waiting periods get rather tiresome, but this back-and-forth use of the Editor and Filer is quite common in practice.

Now press [P], and the following prompt appears:

```
Load what code file?
```

The computer is asking the name of the file containing the program that you want to permanently load. Type:

```
ACCESS:EDITOR.
```
[Return]

to permanently load the Editor (be sure to type the period after `EDITOR`).

Note

If your default volume is set to ACCESS: you can just type the file name; ACCESS: will be assumed.

Press [P] again and type this line to permanently load the Filer:

```
ACCESS:FILER. [Return]
```

Now invoke the Editor, quit it, invoke the Filer, quit it, etc. Notice how quick the response time is now. You can even remove the ACCESS: volume, because the subsystems are permanently loaded in memory, and need not be read from the disc each time they're invoked.

Version

The **Version** command is invoked by pressing [V]. It is used primarily to reset the time and date.

Pressing [V] will recall the Pascal system's initial display and repeat the time and date prompts. You are familiar with these, so let's look at the other information on the screen.

```
New system date ?

  System date is          15-Feb-87
  Clock time is            0: 8:39
  Time zone is             0: 0: 0

  Workstation             Rev. 3.2  15-Jan-87

  Available Global Space 62098 bytes
  Total Available Memory 733688 bytes

  System  volume:  SYSVOL:
  Default volume:  BOOT:



Copyright 1987 Hewlett-Packard Company
All rights are reserved.  Copying or other
reproduction of this program except for archival
purposes is prohibited without prior
written permission of Hewlett-Packard Company.
```

Below the time and date is a line (beginning with `Workstation`) that gives the version of the Pascal Workstation you're using, and the date this version was released. The current version is Rev 3.2.

Next, the amount of available memory in your computer is displayed. The amount of `Available Global Space` requires more knowledge of the Pascal system than is addressed in this book, so you are referred to the *Pascal Workstation System Manual* if interested. The `Total Available Memory` is the amount of memory available for programming after the Pascal system's core programs and all permanently loaded programs have been

loaded. Note that permanently loaded programs do consume memory, so if memory is in short supply, permanently load only the essential subsystems.

Finally, the current settings for the system and default volumes are displayed, along with copyright information.

Press [Return] twice to skip past the date and time prompts and return to the Main Command Level.

Memvol

The **Memvol** command is invoked by pressing [M]. It is used to create a volume, not on a disc, but in your computer's random access memory (RAM).

The advantage to using a memory volume is that it is faster than a disc volume; storing files to, and retrieving files from, a memory volume is virtually instantaneous.

The disadvantage of a memory volume is that, like RAM, it is vulnerable to power loss. If the power should fail, all files stored in a memory volume would be lost.

The best application for a memory volume is during a development cycle, when you are frequently modifying a file. The use of memory volumes will be demonstrated further in Chapter 7, and in Part III of this book.

To create a memory volume, press [M] to get the following prompt:

```
*** CREATING A MEMORY VOLUME ***

What unit number?
```

The computer is asking you to specify which unit number you would like to assign to the memory volume. Unit #50 is a good choice, so type:

```
#50: [Return]
```

Next, the computer asks how much memory (in increments of 512 byte blocks) you want to allocate to the memory volume:

```
How many 512 byte BLOCKS?
```

Your response here will depend on your particular requirements and the amount of memory in your computer. In the examples in this book, only a small memory volume (25K bytes) is needed, so type:

```
50 [Return]
```

Finally, the computer asks:

```
How many entries in directory?
```

Like a disc volume, a memory volume has a directory. The directory consumes memory space, so don't make the directory bigger than necessary. Eight directory entries is the minimum number you can have, so type:

```
8 [Return]
```

The computer now creates the memory volume, and reports:

```
#50: (RAM:) zeroed
```

The memory volume has been assigned to unit #50 and has RAM: as its volume ID. To confirm that the memory volume exists, press [F] to enter the Filer, and press [V] to invoke the **Volumes** command. There is now an entry for unit number 50 that looks like:

```
50 # RAM:
```

Press [Q] to return to the Main Command Level.

Newsysvol

The **Newsysvol** command is invoked by pressing [N]. It is used to assign a different volume as the system volume.

To see which volume is currently assigned as the system volume, press [W] to invoke the **What** command. The current system volume ID is given at near the bottom of the display. Note it and press [Q] to exit.

In this example, we will set the system volume to the RAM: volume just created with the **Memvol** command. Press [N] to get the following prompt:

```
What new system unit number?
```

We want to set the system volume to the memory volume, RAM:. The unit number of RAM: is 50, so type:

```
#50: [Return]
```

To confirm that the system volume is now RAM:, invoke the **What** command again, and look at the entry for the system volume. Leave the system volume as RAM: and return to the Main Command Level. You will use this setting in Chapter 7.

This completes your tour of the Main Command Level.

For More Information...

For details about these and other Main Command Level commands, refer to the *Pascal Workstation System Manual.*

Notes:

The Editor 6

The Editor helps you create files. Its commands make it easy to type in a program or document, change it if necessary, and then store it in a file.

In this section, we discuss and demonstrate the more common editing commands.

When to Use the Editor

Whenever you want to type something—whether it be a Pascal program, an office memo, or a computer manual—think of the Editor. It is the only subsystem that will allow you to type freely; all of the others simply accept and execute commands.

Entering the Editor

You are already familiar with this procedure (it was given in Chapter 4), but we repeat it here to refresh your memory. To enter the Editor from the Main Command Level, follow these steps:

Note

If you have just permanently loaded the Editor by following the instructions for the **Permanent** command (in the preceding chapter), you do not need the ACCESS: volume as described in step 2.

---

1. Make sure the command line for the Main Command Level appears on the screen:

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

   Press [Stop] if it doesn't.

2. Insert your working copy of the ACCESS: volume into one of your flexible disc drives.
3. Now press the [E] key to load and run the Editor program.
4. The following display should now appear on the screen:

```
Editor  [Rev 3.2    15-Jan-87]

Copyright 1986 Hewlett-Packard Company.
        All rights reserved.

No workfile found.
File? (<ret> for new file, <stop> exits)
:
```

This display is the "front door" of the Editor; it is asking you if there is an existing file that you want to edit. Since you have none, simply press [Return] to indicate that you want to create a new file. The Editor's command line then appears on the screen:

```
Editor: Adjst Cpy Dlete Find Insrt Jmp Rplace Xchng Zap ?
```

You are now inside the Editor.

Primary Editing Commands

Insrt (Insert)

The **Insrt** command is invoked by pressing [I]. It allows you to insert text—i.e., type freely—inside the Editor.

The command line for **Insrt** shows what keys are active, as well as keys for accepting and escaping (aborting) everything typed during the insert session. The line is decoded as follows:

- `Text` tells you that all typewriter keys are active for inserting text.
- `<bs>, <clr ln>` tells you that you may use the [Back space] key to simultaneously back up and delete typing errors, and use [Clear line] to erase the line containing the cursor (unless it was the first line inserted).
- `<exc> accepts, <sh-exc> escapes>` tells you to press [Select] when you're finished typing to make your text permanent, or [Shift] [Select] to erase everything you've typed since entering insert mode.

To try it out, press [I] and type in the following letter exactly as shown:

```
Dear HP:

I just got my new Hewlett-Packard Series 200/300 Pascal
Workstation.  I hate typing, but the manual insists that I
try out the Editor, so here I am typing this note to you.

So far, I am

Enjoying
Tolerating
Loathing

this manual.  I find it

Too easy to follow.
Just about right.
Too hard to follow.

Please accept my heart-felt

Thanks
Disgust

for making my introduction to the Pascal system

An enjoyable experience.
An endurable experience.
A real nightmare.

                Respectfully yours,
```

Now press [Select] to exit insert mode. Had you pressed [Shift] [Select] instead, the entire letter would have been erased.

Dlete (Delete)

The **Dlete** command is invoked by pressing [D]. It is the opposite of insert, removing every character that the cursor touches.

The command line for **Dlete** provides the following information:

- > is the direction indicator. It tells you which direction the cursor will move when the space bar is pressed. When the indicator is pointing to the right (>), the cursor will move forward when the space bar is pressed. When the indicator is pointing to the left (<), the cursor will move backward when the space bar is pressed. The direction of the indicator is changed by pressing [>] or [<].
- <> tells you that the space bar may be used to move the cursor forward, wiping out characters as it goes.
- `<Moving commands>` tells you that all of the cursor control keys (the knob, the arrow keys, [Back space] and [Return]) may be used to move the cursor.
- `<exc> deletes, <sh-exc> aborts` tells you to press [Select] when you're finished to make your deletions permanent, or [Shift] [Select] to abort your deletions and return the text to its state before the delete operation was begun.

Try this out. Position the cursor somewhere in the list of words:

```
Enjoying
Tolerating
Loathing
```

Choose the word that most closely expresses your attitude and delete the other two. To delete a word, position the cursor under its first letter and press [D]. Then move the cursor forward to erase the word.

Also try moving the cursor back again, and notice that the word reappears. Your deletions are not permanent until you exit delete mode with the [Select] key. Now erase the word again, and press [Select]. The word is now permanently deleted.

Note

This is not quite true. There is a way to get it back, but we won't discuss it until the **Cpy** command section (coming up).

Next, erase the other words in the list that doesn't express your sentiments, but this time, press [Shift] [Select] to exit delete mode. Notice the word is *not* deleted. Press [D], erase it again, and press [Select] this time to make the deletion permanent.

Continue deleting all inappropriate words in the other three lists, using the space bar to move the cursor. Experiment with the direction indicator, also. Press [<], and notice that the cursor moves backward when you hold down the space bar. Press [>], and the cursor moves forward again.

Next, delete the blank lines that separate your chosen words from the rest of the sentences, but leave the chosen words on separate lines. To do this, position the cursor on a blank line and press [D] to enter delete mode. Then press [Return] to move the cursor to the beginning of the following line, and press [Select] to complete the deletion.

If you looked charitably on this manual, your letter should look like this when you're through:

```
Dear HP:

I just got my new Hewlett-Packard Series 200/300 Pascal
Workstation.  I hate typing, but the manual insists that I
try out the Editor, so here I am typing this note to you.

So far, I am
Enjoying
this manual.  I find it
Just about right.
Please accept my heart-felt
Thanks
for making my introduction to the Pascal system
An enjoyable experience.


                              Respectfully yours,
```

Xchng (Exchange)

The **Xchng** command is invoked by pressing [X]. It allows you to make corrections by typing new letters over old ones; i.e., by exchanging one letter for another.

The command line for **Xchng** shows what keys are active, as well as keys for accepting and aborting everything typed during the exchange session. The line is decoded as follows:

- `Text` tells you that all typewriter keys are active for exchanging text.
- `<bs>` tells you that you may use the [Back space] key to simultaneously back up and delete typing errors.
- `<sh-exc> aborts, <exc> accepts` tells you to press [Select] to make your changes permanent, or [Shift] [Select] to undo everything you've done since beginning the exchange.

In the letter, all of your chosen words begin with upper-case letters. This is not proper, so use the **Xchng** command to change the first letter of each to lower-case.

Position the cursor to the first letter of the first chosen word and press [X]. Now type the lower-case equivalent over the original. While you're at it, hold down the [k] key to replace all forward letters on the line with ks. Notice that you cannot replace letters past the end of the line. Now hold down [Back space] and watch the original letters return. Also note that you cannot exchange letters previous to the letter where you began the exchange.

Finally, make sure the first letter of the word is lower-case, and press [Select] to make the change permanent. Repeat this procedure for the first letters of all chosen words (use [Return] to move the cursor down through lines of text). When you're through, your letter should look something like this, depending upon your choice of words:

```
Dear HP:

I just got my new Hewlett-Packard Series 200/300 Pascal
Workstation.  I hate typing, but the manual insists that I
try out the Editor, so here I am typing this note to you.

So far, I am
enjoying
this manual.  I find it
just about right.
Please accept my heart-felt
thanks
for making my introduction to the Pascal system
an enjoyable experience.


                              Respectfully yours,
```

Cpy (Copy)

The **Cpy** command is invoked by pressing [C]. It is used to copy other files or buffer contents (explained below) into your text. This feature makes it possible to copy and duplicate portions of text.

The command line for the **Cpy** command provides the following information:

- `Buffer` tells you to press [B] to copy the contents of the buffer beginning at the position of the cursor.
- `File` tells you to press [F] to copy the contents of a file beginning at the position of the cursor.
- `<esc>` tells you that the copy operation can be aborted by pressing [ESC].

You've created no files yet, but let's do a dry run anyway to see how it works. You would first position the cursor to the point where you want the file copied. For instance, if you wanted to copy the file at the end of your letter, you would position the cursor there.

Next, press [C], then [F] to indicate your desire to **C**opy a **F**ile. The command line changes to:

```
Copy: File[marker,marker] ?
```

The computer is asking for the complete file specification of the file you want to copy, including the volume ID (or unit number) and the file name. If you had a file to copy, you would type in the file specification and press [Return]. Since you don't, just press [Return] to abort the operation.

The copy buffer operation is very useful for "undoing" an inadvertent deletion, and for moving and copying blocks of text. Before you try it out, you must understand the buffer.

The **buffer** is an area of memory where the text of your last insertion or deletion operation is stored. Every time you perform an insertion or deletion, the old contents of the buffer are replaced with the text that you just inserted or deleted.

To see this, position the cursor at the top of your letter, press [D], and press the [Return] key several times to delete several lines of text. Delete the entire letter if you like. Press [Select] to make your deletions permanent. Everything you deleted is now stored in the buffer. Then, before you do anything else, press [C] and then [B] to **C**opy the **B**uffer. The deleted lines are restored.

The preceding demonstration shows how to recover when you delete something by mistake. You must discover and correct your mistake, however, before another deletion or insertion is performed.

Now let's look at another use of copying the buffer: moving text.

Let's say, for some reason, you decide to make the first paragraph of your letter the last. How do you do this, save for retyping the whole paragraph at the end of the letter, and deleting the original paragraph at the beginning? Immediately the buffer should spring to mind.

If you delete the paragraph, you effectively copy it into the buffer. Then you can move the cursor to the desired position and copy the buffer back into its new position. Here's the procedure in detail:

First position the cursor to the very end of the letter and insert a couple of blank lines. To do this, press [I] to enter insert mode, and press [Return] twice to insert two blank lines. Then press [Select] to exit insert mode. This will separate the paragraph you are going to copy from the rest of the text.

Next, position the cursor to the first letter of the first word in the first paragraph ("I"). Press [D], and then press [Return] until the paragraph is deleted. Press [Select] to accept the deletion.

Now position the cursor a few lines below the end of the letter. (You can move the cursor around without affecting the buffer; the buffer is only changed when [I] or [D] is pressed.) Press [C] [B] and the entire paragraph is copied.

Before your finished, copy the paragraph back into its original position by moving the cursor to the top (just above "So") and pressing [C] [B] again. Note that the buffer is unchanged by the last copy operation, allowing you to copy the same paragraph as many times as you wish.

Finish up by deleting the paragraph you copied at the bottom of the letter.

Set environment

The **Set environment** command is invoked by pressing [S]. It allows you to set several characteristics of the editor, including margins, paragraph indenting, line filling, etc.

When you press [S], you are presented with the following four options:

- **Env** tells you that pressing [E] will allow you to set each editing characteristic individually.
- **Mrk** tells you that pressing [M] will allow you to set markers in your text.
- **Prog** tells you that pressing [P] will preset the editing characteristics to values suitable to typing in and editing Pascal programs.
- **Doc** tells you that pressing [D] will preset the editing characteristics to values suitable for typing in and editing letters, memos, manuals and other documents.

We will examine **Set Prog** and **Set Doc** here; for information on the other options, refer to the *Pascal Workstation System Manual.*

Press [S] [P] (Set Prog) to get the following display:

```
>Environment: {options} <sel> or <sp> leaves
    Auto indent   True
    Filling       False
    Left margin   0
    Right margin  78
    Para margin   5
    Command ch    ^
    Token def     True
    Ignore case   False
    Zap markers
    2 bytes used, 728062 available
```

This "menu" shows how the suggested editing characteristics for a program-editing environment. While you're learning the Editor, it is best to just accept these defaults until you feel you need to change them. To accept the default settings, just press the space bar. The Editor is now set in an environment designed for entering and editing Pascal programs. You will use this environment in later chapters to enter and edit a Pascal program.

Now press [S] [D] (Set Doc) to get the following display:

```
>Environment: {options} <sel> or <sp> leaves
    Auto indent   False
    Filling       True
    Left margin   0
    Right margin  78
    Para margin   5
    Command ch    ^
    Token def     False
    Ignore case   False
    Zap markers
    2 bytes used, 728062 available
```

Notice that the values of several editing characteristics have been changed from the previous program-editing settings. This time, change the right margin setting from 78 to 60. To do this:

1. Press [R] to indicate you want to change the Right margin. Notice the cursor jumps to this location and the entry goes blank.

2. Type in your new value for the right margin:

   60

3. Press the space bar to enter the new value.

And again, press the space bar to leave.

Note

For a complete description of all editing environment characteristics, refer to the Editor chapter of the *Pascal Workstation System Manual.*

Margin

The **Margin** command is invoked by pressing [M]. It is active only in the documentation-editing environment, and is used to "clean up" a paragraph left messy by several insertions and deletions. The appearance of a paragraph after the **Margin** command depends upon the settings of the editing characteristics.

Before you see how this works, you must understand the Editor's definition of a "paragraph." A paragraph is any block of text that has a blank line above it and a blank line below it. By this definition, then, the disheveled block of text lying between "so here I am writing this note." and "Respectfully yours," is a paragraph. Let's clean it up with the **Margin** command.

Simply position the cursor anywhere in the paragraph and press [M]. The paragraph is completely restructured. Notice also that the first paragraph is not indented properly, so position the cursor there and press [M] again.

If Something Goes Wrong...

If you receive the following error message, the editing environment is not set for documents.

```
ERROR: Wrong environment <space> continues
```

Press the space bar, then [S] [D], and then the space bar again, and re-try the **Margin** command.

When you're finished, the letter should look something like this:

```
Dear HP:

     I just got my new Hewlett-Packard Series 200/300 Pascal
Workstation.  I hate typing, but the manual insists that I
try out the Editor, so here I am typing this note to you.

     So far, I am enjoying this manual.  I find it just
about right. Please accept my heart-felt thanks for making
my introduction to the Pascal system an enjoyable
experience.

                              Respectfully yours,
```

Exiting the Editor

Use the following procedure to exit the Editor:

1. Make sure the Editor's command line appears on the screen:

   ```
   Editor: Adjst Cpy Dlete Find Insrt Jmp Rplace Xchng Zap ?
   ```

   If it doesn't, press [Return], [Select] or [ESC].

2. Press [Q] to quit the Editor.

3. The following prompt now appears on the screen:

```
>Quit:
  Update the workfile and leave
  Exit without updating
  Return to the editor without updating
  Write to a file name and return
```

   This is a list of all of the ways you can exit the editor. Each option is described below:

- **Update the workfile and leave.** This option will save the text you created in the "workfile." The workfile is a file the Pascal system creates for you on the system volume. It is used as a temporary file for storing programs or documents while they are being developed. Workfiles are discussed in more detail in the next section.
- **Exit without updating.** This option exits the editor without saving the text you created there. It should be used only if you want to discard the work in your last editing session.
- **Return to the Editor without updating.** This option simply returns you to the Editor, at the same point in your text where you exited it.
- **Write to a file name and return.** This option allows you to save your work in a file of your choosing.

4. In this demonstration, we will choose the last option. Press [W] to get the following prompt: `Name of output file (<ent> to return) -->`

Insert an initialized discs into one of your flexible disc drives. Then store the file under the file name LETTER. To do this, type the unit number of the flexible disc drive containing your disc (you should know this from Chapter 3), followed by the file name LETTER. Press [Return]. For example, if you are storing the text on the disc in unit #3, you would type:

```
#3:LETTER [Return]
```

The text is then saved on the disc in the file named LETTER.TEXT. Notice the string of dots moving across the screen, indicating the file is being written to the disc.

The Editor automatically appended the suffix .TEXT to your file, so its full name on the disc is LETTER.TEXT. This suffix tells you that the file has been stored using the *text* format, which includes not only your words, but also the editing environment in force when the file was saved. Since you exited the Editor with the environment set in "Doc" mode, these settings will automatically be restored when you return LETTER.TEXT to the Editor for editing.

It's worth mentioning that you could have also saved this file using the ASCII file format, by specifying the file name as LETTER.ASC rather than as LETTER. The ASCII format saves your file as a string of ASCII characters; no environment information is saved. If you are saving the file on an HFS disc, you could save the file as LETTER.UX and it would then be compatible with the HP-UX Workstation.

5. After the file is written to the disc, the size of the file is displayed (in bytes), and you are asked:

```
Exit from or Return to the editor ?
```

You can press [R] to return to the Editor or [E] to exit to the Main Command Level. Press [E].

You are now back at the Main Command Level. This completes your tour of the Editor.

For More Information...

To learn more about the Editor, refer to the Editor chapter of the *Pascal Workstation System Manual.*

Notes:

The Filer 7

The Filer is the main connection between your computer and its disc drives. It allows you to store, retrieve, delete and copy files. Volume manipulating commands are well represented also, permitting you to list all the files in a volume, rename a volume, compact a volume and even destroy a volume.

When to Use the Filer

Use the Filer whenever you need the services of your disc drives. Or more generally, whenever you want to move programs, data or text from one place to another.

Entering the Filer

Follow this procedure to enter the Filer:

1. Make sure the command line for the Main Command Level appears on the screen:

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

2. Insert your working copy of the ACCESS: volume into one of your flexible disc drives.
3. Now press the [F] key to load and run the Filer subsystem.
4. The following command line should now appear at the top of the screen:

```
Filer: Change Get Ldir New Quit Remove Save Translate Vols What Access Udir ?
```

You are now inside the Filer.

Getting Ready for the Demonstrations

Prerequisites

Before you proceed, make sure you understand the relationships between files, volumes and unit numbers, and are well acquainted with the unit number assignments for your system. Refer back to Chapter 3 if you need a refresher.

Introducing Workfiles

It's important to understand the how workfiles are used in the Pascal system.

On the surface, workfiles are like any other file. One workfile is a text file, named WORK.TEXT, and the other is a code file, named WORK.CODE. Both are stored on the system volume. What makes the workfiles so special is the way the Pascal subsystems use them.

In the Editor, the initial display looked like this:

```
Editor  [Rev 3.2   15-Jan-87]

Copyright 1987 Hewlett-Packard Company.
         All rights reserved.

No workfile found.
File? (<ret> for new file, <stop> exits)
:
```

Notice the fourth line of this display says:

```
No workfile found.
```

The Editor looked first to see if there was a workfile (i.e., a file named WORK.TEXT on the system volume). Because it didn't find it, it asks you for the name of the file you want to edit. If it *had* found the workfile, it would have *automatically* loaded it. The Editor assumes you want to edit the workfile, if it exists.

The Compiler works much the same way. If it finds the workfile named WORK.TEXT, it assumes you want to compile it. If the compilation is successful, it updates the workfile WORK.CODE with the compiled code of WORK.TEXT.

So the workfiles are like the system "default" files; if they exist, the subsystems assume you want to use them. The advantages to using workfiles will become apparent as you work through this chapter, as well as in Part III, when you will use workfiles in a programming example.

Setting the System Volume

Many of the demonstrations in this chapter involve workfiles. Since workfiles are stored on the system volume, it's important to have the system volume set correctly before trying out our examples.

We will use the small memory volume you created in the Chapter 5 as the system volume. If you followed the instructions for the **Memvol** and **Newsysvol** commands in Chapter 5, then you have already created a memory volume at unit #50 and have assigned it as the system volume. Here's how to check:

Enter the Filer, then press [V] to invoke the **Volumes** command. If there is an entry for unit number 50 that looks like this, you have already created your memory volume:

```
50 * RAM:
```

If there is no entry for unit number 50, go back to Chapter 5 and make a memory volume, using the instructions for the **Memvol** command. If the asterisk (*) does not appear between 50 and RAM:, then the memory volume is not assigned as the system volume. Set the memory volume to the system volume using the instructions for the **Newsysvol** command.

Primary Filer Commands

Get

The **Get** command is invoked by pressing [G]. It is used to associate a file name with the workfile. When you "get" a file, it is like *copying* it into the workfile, where it then becomes the "default" file for all Pascal subsystems. Try this exercise:

1. Find the disc containing the file LETTER.TEXT that you created in Chapter 6. Insert it into unit #3.
2. Press [G] to invoke the **Get** command. The computer responds with:

   ```
   Get what file?
   ```

3. Type the following response to assign the file LETTER.TEXT in unit #3 to the workfile WORK.TEXT:

   ```
   #3:LETTER
   ```

4. The computer finds the file LETTER.TEXT in unit #3 and reports:

   ```
   Source file loaded
   ```

5. Unfortunately, the last statement the computer made is not quite true. It leads you to believe that the source file, LETTER.TEXT, has been copied into the workfile, WORK.TEXT. You can see this is not true by listing the directory of the system volume:

   a. Press [L] to invoke the **List** command. When the computer asks:

      ```
      List what directory?
      ```

      type the following to list the system volume (remember, the special symbol * represents the system volume):

      ```
      * [Return]
      ```

   b. The computer displays the listing of the *empty* directory of the system volume RAM:

      ```
      RAM:                Directory type= LIF level 1
      created  9-Apr-86 16.56.24 block size=256
       Storage order
      ...file name....     # blks     # bytes  last chng

      FILES shown=0 allocated=0 unallocated=8
      BLOCKS (256 bytes) used=0 unused=97 largest space=97
      ```

WORK.TEXT, the workfile, is not there.

6. The file LETTER.TEXT will not actually be copied into WORK.TEXT until the workfile is *updated*—one of the exit options of the Editor. Press [Q] to quit the Filer, and press [E] to enter the Editor.

7. Notice that the Editor didn't prompt you for the name of a file to edit; it simply read in LETTER.TEXT, which has been identified as the workfile. Press [Q] to quit the Editor, and then [U] to select "Update the workfile and leave" as your exit option.

8. Now press [F] to re-enter the Filer, and list the system volume again as you did in step 5. The directory now contains the updated workfile, WORK.TEXT:

```
RAM:                Directory type= LIF level 1
created  9-Apr-86 16.59.10 block size=256
 Storage order
...file name....     # blks     # bytes  last chng

WORK.TEXT                   8        2048  9-Apr-86
FILES shown=1 allocated=1 unallocated=7
BLOCKS (256 bytes) used=8 unused=89 largest space=89
```

Save

The **Save** command is invoked by pressing [S]. It is used to save the workfile WORK.TEXT into another, and usually more permanent, file.

Save is the opposite of **Get**. If you use **Get** to copy a file into WORK.TEXT, you can use **Save** to copy WORK.TEXT back into its original file. Try it with your current workfile:

Press [S] to invoke the **Save** command. The computer responds with:

```
Save as #3:LETTER.TEXT ? (Y/N)
```

The computer remembers that your current workfile was retrieved from the file LETTER.TEXT in unit #3, so it's asking if you want to save it back in the same file. Respond by typing [Y].

The computer then responds with:

```
#3:LETTER.TEXT
exists ... Remove/Overwrite/Neither ? (R/O/N)
```

Since the file LETTER.TEXT already exists in the volume in unit #3, the computer wants to know what you want to do with the old file. You obviously don't want two files with the same name on the same volume, so your options are:

- Press [R] to first remove the old file, then write the new file.
- Press [O] to write the new file directly over the old one. See below.
- Press [N] to abandon the **Save** operation altogether.

Press [R] to remove the old file and write the new one.

Use the `Overwrite` option only on hierarchical file systems, or if the new version of the file is not larger than the old version. On HSF and SRM systems, this option updates the file such that all duplicate links to the file point to the updated copy of the file. (In contrast, choosing the `Remove` option updates the file such that the path you specify points to the updated version of the file, but any other duplicate links to the file remain set to the original (old) copy of the file.)

Ldir

The **Ldir** command is invoked by pressing [L]. It is used to obtain a list of all files in a specified directory, including important information about each file.

Insert the ACCESS: volume into unit #3 and press [L] to get the following prompt:

```
List what directory?
```

Type:

```
#3: Return
```

The directory of the ACCESS: volume is then displayed, and should resemble this (if you have double-sided micro discs, the directory will contain additional files not shown below):

```
ACCESS:            Directory type= LIF level 1
created 14-Nov-86 10.46.18 block size=256
 Storage order
...file name....    # blks    # bytes  last chng

FILER                  248       63488 14-Nov-86
EDITOR                 234       59904 14-Nov-86
LIBRARIAN              292       74752 14-Nov-86
MEDIAINIT.CODE         144       36864 14-Nov-86
TAPEBKUP.CODE           70       17920 14-Nov-86
FILES shown=6 allocated=6 unallocated=2
BLOCKS (256 bytes) used=988 unused=65 largest space=65
```

There is a lot of information in the directory, so let's pick it apart, line by line:

First Line:

```
ACCESS:              Directory type= LIF level 1
```

This tells you the name of the volume whose directory this is, namely ACCESS:. It also tells you the directory type, which may or may not interest you. If it does, refer to the *Pascal Workstation System Manual.*

Second Line:

```
created 14-Nov-86 10.46.18 block size=256
```

This line tells you the date and time when the volume was created, namely April 4, 1986 at 33 seconds past 3:07 in the afternoon (your ACCESS: volume may have a different inception date). It also tells you that a **block** of disc storage is equal to 256 bytes. A block is merely a unit of disc storage space; file sizes are expressed in the number of blocks they occupy on the disc.

Third Line:

```
Storage order
```

This line tells you that the files are listed in the order in which they are stored on the disc. The alternative is alphabetic order, discussed in a moment.

Fourth Line:

```
...file name...    # blks    # bytes   last chng
```

This is the heading for the list of files that follows. The `file name` is, of course, the name of the file. The `# blks` is the number of blocks of disc storage occupied by the file, a measure of its size. The `# bytes` is the number of bytes of disc storage occupied by the file. Since a block on a LIF disc is 256 bytes, this number is just the previous column multiplied by 256. The `last chng` column tells when the file was last changed; i.e., read from, and written back to the disc.

Note

The dates displayed in the directory are read from the system time and system date that you set after booting the Pascal system. This is one good reason to keep the time and date current.

Lines 5 through 10

```
FILER                248      63488 14-Nov-86
EDITOR               234      59904 14-Nov-86
LIBRARIAN            292      74752 14-Nov-86
MEDIAINIT.CODE       144      36864 14-Nov-86
TAPEBKUP.CODE         70      17920 14-Nov-86
```

These are the files on the ACCESS: volume listed in the order in which they are stored.

Line 11

```
FILES shown=6 allocated=6 unallocated=2
```

This line tells you that six files are **shown** in this listing; that there are six entries **allocated** in the directory, and that two entries remain **unallocated**.

Recall when you initialized your discs, one of the prompts was for the number of directory entries. The number you entered was the maximum number of files that could be stored on that volume. Here, the listing is showing you how many of those directory entries have been filled by files, and how many are left before the directory (and hence the volume) is full. Understand that you can run out of directory entries but still have room left on your disc. Volume storage management will be discussed more in the **Krunch** command section of this chapter, and in Part III.

Line 12

```
BLOCKS (256 bytes) used=988 unused=65 largest space=65
```

The preceding line told you how much *directory* space was allocated on the volume; this line describes how much *file storage* space remains.

The **used** field tells you how many 256-byte blocks of disc storage have been used by the files listed. **unused** tells you how many blocks remain before the disc runs out of file storage space. The **largest space** gives the largest contiguous "hole" in the unused storage space. If you try to write a file to a LIF disc that is larger than the largest space, the file cannot fit in this hole and thus cannot be written. The **Krunch** command can be used to widen this hole, however, as described later.

One final note before we move on: you can list the files in alphabetic order by typing [*] after the unit number or volume ID. To list the ACCESS: disc in alphabetic order, press [L] and then type:

#3:[*] [Return]

The alphabetical listing produced should resemble this:

```
ACCESS:            Directory type= LIF level 1
created 14-Nov-86 10.46.18 block size=256
 Storage order
...file name....    # blks    # bytes  last chng

EDITOR                 234      59904 14-Nov-86
FILER                  248      63488 14-Nov-86
LIBRARIAN              292      74752 14-Nov-86
MEDIAINIT.CODE         144      36864 14-Nov-86
TAPEBKUP.CODE           70      17920 14-Nov-86
FILES shown=6 allocated=6 unallocated=2
BLOCKS (256 bytes) used=988 unused=65 largest space=65
```

Note

If a listing is too long to fit on the screen, the listing will pause after displaying the first few files. Press the space bar to see the remaining files.

Filecopy

The **Filecopy** command is invoked by pressing [F]. It is used to make copies of files (or entire volumes).

You used **Filecopy** to back up your Pascal system in Chapter 3. Refer there for procedures for copying entire volumes. In Part III of this manual, we will demonstrate how to use **Filecopy** to copy multiple individual files to a Winchester disc. Here we will demonstrate how to copy a single file.

In this example, we will copy the file LETTER.TEXT in unit #3 to the RAM: volume (unit #50).

Press [F] to invoke the **Filecopy** command. The computer displays the following prompt:

```
Filecopy what file ?
```

Type:

```
#3:LETTER.TEXT [Return]
```

The computer then asks:

```
Filecopy to what ?
```

Type:

```
#50:$ [Return]
```

The **$** is a *wildcard.* It tells the computer to use the same file name for the new copy of the file. Thus, it is equivalent to typing **#50:LETTER.TEXT**, but is much more convenient.

The computer copies the file from unit #3 to unit #50, and gives the following message when completed:

```
V3:LETTER.TEXT                    ==> RAM:LETTER.TEXT
```

Change

The **Change** command is invoked by pressing [C]. It is used to change the name of a file.

To see how it works, change the name of LETTER.TEXT in unit #50 to MAIL.ASC:

Press [C] to get the following prompt:

```
Change what file?
```

Type in the location and name of the file you want to change:

```
#50:LETTER.TEXT [Return]
```

The computer then asks:

```
Change to what?
```

Type:

```
MAIL.ASC [Return]
```

Note

Type in only the new file name; *do not* type in the unit number again. You are only changing the name of an existing file, not moving it to another volume. Thus, the location of renamed file is the same as the location of the original file.

The computer confirms the change with:

```
RAM:LETTER.TEXT              ==> RAM:MAIL.ASC
```

You may further confirm the change by listing the directory of the RAM: volume, if you wish. Just press [L], then type:

```
#50: [Return]
```

Remove

The **Remove** command is invoked by pressing [R]. It is used to remove a file from a volume.

Use **Remove** to delete the file MAIL.ASC from the RAM: volume (unit #50). Press [R] to get the following prompt:

```
Remove what file?
```

Type the location and name of the file to be removed:

```
RAM:MAIL.ASC [Return]
```

The computer responds with:

```
RAM:MAIL.ASC removed
```

Confirm that MAIL.ASC is no longer on the RAM: volume by listing it as you did in the last demonstration.

Krunch

The **Krunch** command is invoked by pressing [K]. It is used to compact all files in a volume into one contiguous block, putting all unused storage space in one contiguous "hole" at the end.

After storing, deleting and updating files on a volume several times, the storage space becomes "fragmented": chopped into small, discontinuous segments. Eventually, you cannot write another file to the volume because there is not a single hole available that is large enough to contain it. The computer will display the message:

```
No room on volume
```

The solution is to **Krunch** all of the files to the front of the volume, joining all of the free space into one large hole at the back.

Note

You can **not** and need not use this command on HFS or SRM systems.

To demonstrate the **Krunch** command, insert the ACCESS: volume into unit #3 and press [K]. The following prompt will appear:

```
Crunch what directory ?
```

Type in the name of the volume to be compacted; in this case:

```
ACCESS: [Return]
```

The computer asks again:

```
Crunch directory ACCESS ? (Y/N)
```

Yes, you're sure, so press [Y]. The operation begins, and the computer displays:

```
Crunch of directory ACCESS in progress
DO NOT DISTURB!!
```

There is a reason why the computer is so emphatic about this. If you were to open the disc drive door, remove the disc, or lose power during this operation, some, if not all, of your files could be lost.

When the crunch is completed, the computer reports:

```
Crunch completed.
```

If you are ever refused when trying to write a file to a LIF or WS1.0 volume, write the file temporarily to a different volume, crunch the volume that refused you, then **Filecopy** the file from the other volume.

Note

Memory volumes, like most others, need crunching too.

Prefix-vol

The **Prefix-vol** command is invoked by pressing [P]. It is used to change the prefix volume (default volume).

The prefix volume is another name for the default volume. The Filer calls it the prefix volume; the **What** and **Version** commands in the Main Command Level call it the default volume.

To "prefix" to a different volume (i.e., reset the default volume), use the **Prefix-vol** command while inside the Filer, or use the **What** command at the Main Command Level.

To use **Prefix-vol**, press [P] to get the following prompt:

```
Prefix to what directory ?
```

Insert the ACCESS: disc in unit #3: and type:

```
#3: [Return]
```

The computer responds with:

```
Prefix is ACCESS:
```

To prefix to a *unit number* rather than to the volume residing in the unit, follow this same procedure, but *do not insert a disc into unit #3.* The prefix volume is then set to unit #3, regardless of what volume is present. The advantage of this is that you can swap discs in and out of unit #3 and never type the volume ID to access a file; whatever disc resides in unit #3 is effectively the default volume. If you are prefixed to a particular volume ID, however, you must change the prefix volume every time you swap discs (because you are prefixed to a single volume).

Vols

The **Volumes** command is invoked by pressing [V]. It is used to list all volumes that are on-line.

You have already used the **Volumes** command several times, but let's examine it more carefully now. Put discs in all of your disc drives and press [V]. The display will look something like this one, depending on your particular system configuration:

```
Volumes on-line
  1   CONSOLE:
  2   SYSTERM:
  3 * ACCESS:
  4 # V3:
  6   PRINTER:
Prefix is - ACCESS:
```

The left-hand column lists all unit numbers; the right-hand column lists each unit number's associated volume ID. The # between the columns indicates a *blocked* device; i.e., a device capable of storing files, such as a disc drive or a bubble memory card. The volume marked with an asterisk (*) is the system volume; the prefix (default) volume is given at the bottom.

Certain unit numbers are permanently assigned. These include:

- Unit #1, volume ID CONSOLE:. This is your computer's screen.
- Unit #2, volume ID SYSTERM:. This is your computer's keyboard.
- Unit #6, volume ID PRINTER:. This is your printer.

Note that all of these are unblocked devices; they cannot store files. Saying that they have a volume ID is rather strange, therefore, since we described volumes earlier as a collection of files. The Pascal Workstation uses this convention for the sake of consistency, so that all devices may be referenced similarly (by unit number or volume ID). Just think of unblocked volume IDs as simply identifiers that allow you to access the devices.

New

The **New** command is invoked by pressing [N]. It is used to clear the workfiles, WORK.TEXT and WORK.CODE.

If there wasn't some way of clearing the workfiles, you would have to access the same file over and over again. The Editor would always assume you wanted to edit WORK.TEXT, and would read it in automatically. The only way around this is to clear the workfile, and force the Editor to ask you for the name of the file you wish to edit.

To clear the workfiles, just press [N]. If you haven't used **Save** to permanently store the workfile, you may get the following message:

```
Throw away current workfile ? (Y/N)
```

Answer with [Y], and the workfile is cleared.

To confirm that the workfile is gone, list the directory of the RAM: volume. Notice that WORK.TEXT has been removed.

Translate

The **Translate** command is invoked by pressing [T]. It is used to translate one file type into another, and to print files on the printer.

To see how **Translate** works, we'll send LETTER.TEXT to the printer. Make sure the disc containing the file LETTER.TEXT is inserted into unit #3 and press [T] to get the following prompt:

```
Translate what file ?
```

Type:

```
#3:LETTER.TEXT [Return]
```

The computer then asks:

```
Translate to what ?
```

Recall from the **Volumes** command that the unit number for the printer connected to your workstation is #6:. Type:

```
#6: [Return]
```

The computer translates LETTER.TEXT into a format suitable for the printer.

Note

If your printer failed to print LETTER.TEXT, make sure it is turned on and properly connected to the computer. If it has an "on-line" light, make sure it is on. If the printer still doesn't work, refer to the "Special Configurations" chapter of the *Pascal Workstation System Manual.*

If your workstation is on an SRM system, translate the file into an ASCII-type file in a shared printer's spooler directory. (Ask your SRM system manager for the name of each shared printer's spooler directory.) For example, type:

```
#5:/lp/FFLETTER.ASC Return
```

The computer translates LETTER.TEXT into an ASCII format for printing on a shared printer and places the translated copy of the file in the printer's spooler directory ("lp" in this example). The "FF" at the beginning of the file name prevents the printer from writing over paper perforations.

Additional Filer Commands

Although not used as frequently as the commands discussed in the previous section, there are several other Filer commands worth noting. These commands are described completely in the Filer chapter of the *Pascal Workstation Manual.*

Ext-dir

This command provides an extended listing of the files in the directory. The extended listing displays additional information but uses two lines for each file instead of just one.

What

The What command prints the name and status of the work file.

Zero

This command is used to place an empty directory on the specified volume. Generally, it is used after a disc has been initialized and before any files are stored on it. The command will not work on HFS or SRM discs.

CAUTION

Possible loss of data: Zeroing a disc will result in all existing information being destroyed.

Bad-secs

The Bad Sectors command can be used to check the readability of each sector on a disc. It should only be used if you believe you have a faulty disc.

Duplicate

This command is used in HFS and SRM file systems to allow a file to be listed in more than one directory. Unlike copying the file — which would take twice as much space on the disc — only the file name is copied to the new directory.

Udir

The Unit Directory command is similar to the Prefix command but does not change the default volume. It is used with HFS and SRM discs.

Access

This command allows you to set passwords on SRM files and directories. See Appendix D or the *Pascal Workstation Manual* for information on setting SRM access passwords.

Hfs

This command allows you to set the permission of files and directories on Hierarchical File System (HFS) discs. See Appendix E or the *Pascal Workstation Manual* for information on setting permissions.

This concludes our tour of the Filer.

Exiting the Filer

Use the following procedure to exit the Filer:

1. Make sure the Filer's command line appears on the screen:

```
Filer: Change Get Ldir New Quit Remove Save Translate Vols What Access Udir ?
```

   If it doesn't, press [Return] or [N].

2. Press [Q] to quit the Filer.

You are now back at the Main Command Level.

For More Information...

For details on these and other Filer commands, refer to the Filer chapter of the *Pascal Workstation System Manual.*

The Compiler 8

Computers are really very simple creatures. They have an extremely limited vocabulary and get very indignant when you talk to them in a language they cannot understand.

Pascal is a language that computers cannot understand. Your computer cannot directly execute Pascal programs created in the Editor. Programs must first be compiled (translated) into a sequence of machine language instructions. The machine language instructions are put in a separate file (a *code* file). The computer then executes the code file.

This translation process is performed by the Compiler.

When to Use the Compiler

After creating a Pascal program in the Editor, use the Compiler to compile the program before you run it. If you make changes to the program, be sure to recompile it, for the changes must be translated to the code file before they have an effect.

Getting Ready for Chapter 8

Before you can see how the Compiler works, you need a program to compile. Use the following procedure to create a simple Pascal program:

1. Press [E] to enter the Editor, and press [Return] to create a new file. If the Editor automatically reads in a workfile, exit the Editor, enter the Filer, and use the **New** command to clear the workfile. Then return to the Editor.
2. Press [I] to enter insert mode and type the following program that prints "HELLO" on the screen:

```
PROGRAM TEST (INPUT,OUTPUT);
BEGIN
  WRITELN ('HELLO')
END.
```

3. Press [Select] to exit insert mode. Press [Q] to quit the Editor, and then press [W] to get the following prompt:

```
Name of output file (<ret> to return) -->
```

4. Insert an initialized disc into unit #3. Type the following response to write the program into a file named TEST.TEXT in the volume in unit #3:

```
#3:TEST [Return]
```

5. Press [E] to exit to the Main Command Level

Entering the Compiler

Follow this procedure to enter the Compiler:

1. Make sure the command line for the Main Command Level appears on the screen:

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

2. Insert your working copy of the CMP: volume into one of your flexible disc drives.
3. Now press [C] to load and run the Compiler subsystem.
4. The following display should now appear on the screen:

```
Pascal [Rev 3.2  1/15/87]   15-Feb-87  08:41:07

     Copyright 1987 Hewlett-Packard Company.
              All rights reserved.

Compile what text?
```

You are inside the Compiler. If you have only one flexible disc drive, remove the CMP: volume now and insert the disc containing the file TEST.TEXT.

Using the Compiler

Notice that the bottom line of the display is a prompt:

```
Compile what text?
```

The Compiler needs to know the name and location of the file containing the program you want to compile. To compile the TEST program that you just created, type:

#3:TEST [Return]

If Something Goes Wrong...

The Compiler may not find your file and will report:

```
not found.  file?
```

Retype the file specification, making sure that you include the correct volume ID or unit number, and the exact file name, including the suffix if other than .TEXT. If you still have problems, press [Return] to exit the Compiler, and use the Filer's **Ldir** command to make sure the file is in the volume. Also check the file name.

The computer assumes that TEST is a text file, and looks for a file named TEST.TEXT in unit #3. It finds it, and asks the next question:

```
Printer listing (l/y/n/e) ?
```

A printer (program) listing shows the complete text of the program, with all lines numbered and all compiler errors marked. You may respond to this prompt in any one of the following ways:

- [l] allows you to send the program listing to a file that you specify.
- [y] sends the program listing to your printer.
- [n] does not produce a program listing.
- [e] sends a program listing to the printer *only* if compiler errors occur.

In this demonstration, we will send the listing to a file. Press [L] to get the following prompt:

```
What listing file?
```

Type the following response to send the listing to a file named TESTLIST.TEXT in unit #3:

```
#3:TESTLIST[25] [Return]
```

The `[25]` specifies the size of the listing file (in blocks). A full listing page requires approximately 25 blocks. The listing for this program will be less than one page, so 25 blocks is plenty of room.

There is one last step before the program is compiled. The *output* (final product) of the Compiler is a code file containing the machine language version of the program text. The Compiler must know the name of the *output file* where you would like the code put. It asks:

```
Output file (default is "#3:TEST.CODE") ?
```

The Compiler will, by default, put the code in a file named TEST.CODE in the volume in unit #3. This is a good choice, because it makes sense to have the code for TEST.TEXT in a file named TEST.CODE. Press [Return] to accept the default file name.

The Compiler now reads in TEST.TEXT and begins compiling it. Progress is reported at the bottom of the screen:

```
<    0>
TEST                [521258]
<    2>

4 lines, No errors.  No warnings.
```

The last line indicates no compiler errors were found. Had there been errors, the line number where the error occurred and the error type would have been reported. No code file would have been generated. Part III will demonstrate how the Compiler reacts to errors.

When compilation finishes without errors, the system immediately exits the Compiler and returns to the Main Command Level.

Press [F] to enter the Filer, and press [L] to invoke the **Ldir** command. The following prompt will appear:

```
List what directory?
```

Type the following response to list the directory of the volume in unit #3:

```
#3: [Return]
```

Notice the two new entries for TEST.CODE and TESTLIST.TEXT.

Quit the Filer and enter the Editor. When prompted for the file you wish to edit, type the following response to view the listing file for program TEST:

```
#3:TESTLIST [Return]
```

The listing looks like this:

```
Pascal [Rev 3.2  1/15/87] TEST.TEXT              15-Feb-87 08:54:39 Page 1

     1:D        0 PROGRAM TEST (INPUT,OUTPUT);
     2:C        1 BEGIN
     3:C        1   WRITELN ('HELLO')
     4:C        1 END.

No errors. No warnings.
```

For instructions on reading the listing, refer to "Interpreting the Compilation Listing" in the Compiler chapter of the *Pascal Workstation System Manual.*

Press [Q] to quit the Editor, then [E] to exit without updating.

Running a Compiled Program

To run a program after compiling it, return to the Main Command Level and press [R] to invoke the **Run** command. Do this now, and notice that `HELLO` is printed on the screen. This is the output of program TEST. To run the program again, you can either press [R] to invoke **Run**, or press [U] to invoke the **User** restart command.

To run a program that is *not* the last one compiled, use **eXecute** to load and run the code file, as you did when initializing discs.

Compiling a Workfile

Compiling a workfile is a little different than compiling a conventional file. To see this, enter the Filer and press [G] to invoke the **Get** command. The following prompt will appear:

```
Get what file?
```

Type the following line to assign the code and text files for program TEST as your workfiles:

```
#3:TEST [Return]
```

The computer responds with:

```
Source and Code file loaded
```

Notice that both source (TEST.TEXT) and code (TEST.CODE) files were loaded with this single command. This is another advantage of giving both your text and code files the same file name, using the suffix (.TEXT or .CODE) to distinguish them.

Exit the Filer and enter the Compiler. Since you now have a workfile, the Compiler assumes it is the file you want to compile. Therefore, the Compiler skips directly to the "printer listing" prompt, and does not ask what text you want to compile.

Note

If a workfile exists, it is impossible to compile any other file. You must clear the workfile with the Filer's **New** command before you can again choose which file you want to compile.

Press [N] to generate no printer listing. Compilation then begins immediately. Notice that the "output file" prompt was also skipped. Because you are using workfiles, not only is the Compiler's *input* file assumed, but the *output* file as well.

So compiling workfiles saves you from responding to two Compiler prompts, which can be very helpful during long program development sessions. On the other hand, if you want to make your own choices about which file to compile and where to direct the output, workfiles should not be used.

Press any key to clear the screen.

Notes:

Other Subsystems 9

The Pascal system includes three other subsystems that will not be demonstrated in this manual. You may have noticed them in the Main Command Level's **What** command display. These subsystems—the Assembler, Librarian and Debugger—require a more advanced knowledge of programming, and thus exceed the scope of this book. We mention them briefly here, to provide a general idea of their functions, and refer you to the *Pascal Workstation System Manual* for details.

The Assembler

In certain applications, the execution time of a program segment is critical. While the Pascal Compiler is quite proficient at compiling a source program into very efficient machine code, time-critical applications may require the superior skills of a human being.

The Assembler is included in the Pascal Workstation for this purpose. It allows you to write MC68000 Processor Assembly language programs to optimize critical sections of a program.

If your applications require assembly language optimization, refer to the Assembler chapter of the *Pascal Workstation System Manual*, and to the *MC68000 User's Manual.*

The Librarian

The Pascal Language System allows you to create "libraries" of compiled programs. New programs that you write can "borrow" programs from these libraries. Wise use of libraries can save you the time and trouble of duplicating a subprogram in several different programs that require it.

The Librarian subsystem helps you create and manage program libraries. For details on the Librarian, refer to the Librarian chapter of the *Pascal Workstation System Manual.*

The Debugger

The Debugger subsystem has a number of features that help you find errors in your programs. It allows you to "step" through program execution one statement at a time, trace program flow, check the values of program variables, etc.

The Debugger is also a good protection against inadvertently restarting the booting process by accidentally pressing [Reset]. It serves as a simple integer calculator, as well.

For more information on the Debugger, refer to the Debugger chapter in the *Pascal Workstation System Manual.*

Backup Utilities

The **BACKUP** program provided with the Pascal Workstation allows you to copy (backup) any or all of your files. See Appendix F before using the BACKUP program. Another backup program, called `TAPEBKUP` is provided for discs that have built-in tape drives. Both programs are described in the *Pascal Workstation* manual.

HFS Support

The Pascal Workstation now supports a local Hierarchical File System. There are several utilities and modules you may wish to know about if you choose to use HFS.

MKHFS — this program installs an HFS directory structure on an initialized disc. (You still use `MEDIAINIT` to initialize the disc.)

HFSCK — this program checks the integrity of the file system and allows interactive repair of the structure if necessary.

OSINSTALL — this program installs a system (bootable) file in the proper location of an HFS disc to allow booting the workstation from an HFS disc. This allows the BASIC, Pascal, and HP-UX workstations to boot from the same hard disc.

HFS_DAM — this module is required to read/write an HFS disc.

UXTEXT_AM — this module enhances the reading and writing of HP-UX text files (type `.UX` files).

For more information about HFS, see Appendix E of this manual.

Notes:

Setting Up Your Environment 10

When you get ready for a long drive in the car, you adjust the seat, align the mirrors, check the oil and tires, and generally prepare your travelling environment to be as safe and comfortable as possible. As you get ready for long programming sessions with the Pascal Workstation, it also makes sense to set up a *programming environment* that is safe and comfortable.

This chapter assumes that you have set the time and date as shown in Chapter 3 and shows you how to:

- Copy part of the Pascal system to a Winchester disc (if you have one)
- Boot the Pascal system from a Winchester disc (if you have one)
- Permanently load the Editor and Filer
- Make a memory volume
- Create an "Autostart" file that automatically sets up many aspects of your programming environment every time you boot the system
- Set the system and default volumes

This chapter assumes that you have already set the date and time as described back in Chapter 3.

Setting Up Your Winchester (Hard) Disc

This section tells you how to copy part of the Pascal system to a Winchester (or other hard) disc if you are going to be using the LIF file system. If you do not have a Winchester disc, skip this section. If you are going to the Winchester disc as a Hierarchical File System (HFS) disc, see Appendix E.

Winchester Disc Considerations

Before you begin storing files on your Winchester disc, you should read the Special Configurations chapter of the *Pascal Workstation System Manual.* There you will learn how to connect multiple Winchester discs to your system, and how to change the way the Pascal system "partitions" your disc into volumes.

For the moment, however, finish reading this manual to gain a deeper understanding of how the Pascal subsystems work together. Although this section will show you how to copy a part of the Pascal system to a Winchester disc, understand that these instructions are for demonstration purposes only. You should not consider your disc fully operational until you have at least looked over the Special Configurations chapter.

Copying the Boot Files

You probably bought a Winchester disc to gain some independence from the constant disc swapping that occurs when using flexible discs. Copying the contents of the BOOT: (or BOOT2:) volume to the first volume of your Winchester disc will allow you to boot the Pascal system directly from the Winchester disc.

Note

The 3.0L, 2.0 and 1.0 versions of the Series 200 Boot ROM cannot boot from a Winchester disc. If your computer has one of these Boot ROMs, do not copy the BOOT: volume to the Winchester disc. You can, however, copy the Pascal subsystems to your Winchester, as described in the next section.

This procedure, like the rest of the book, assumes you have only one Winchester disc on-line.

1. The following line should appear at the top of the screen:

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

2. Insert the disc labeled ACCESS: into a flexible disc drive. Press the [F] key to load the Filer. Wait for the Filer's command line to appear at the top of the screen:

```
Filer: Change Get Ldir New Quit Remove Save Translate Vols What Access Udir ?
```

3. Remove the ACCESS: volume and insert the BOOT: (or BOOT2:) volume.
4. Press [F] to invoke the **Filecopy** command. We will use the **Filecopy** command to copy all of the files from the BOOT: volume to the Winchester disc volume assigned to unit #11.
5. The following prompt now appears on the screen:

   ```
   Filecopy what file ?
   ```

   The computer is asking for the file specification of the *source* file (i.e., the file you want to copy). You could copy each file in the BOOT: volume separately, using several invocations of the **Filecopy** command, but it would be much more convenient to copy them all at once. Do this by using the = wildcard. Type:

   `BOOT:=` [Return]

   The = wildcard says to copy all of the files on the BOOT: volume, but do so one at a time. It's important to understand the distinction between this operation and copying an entire *volume* of files, as you did when making back-up copies on flexible discs. If you left off the wildcard and typed `BOOT:` as your response, the volume on unit #11 would be replaced by a copy of the BOOT: volume. What's wrong with that? Nothing, except that unit #11, which contains one megabyte or more of mass storage, would suddenly become the size of the BOOT: volume on the flexible disc, which contains slightly over 256K bytes of mass storage. Thus, a volume copy would render most of the storage capacity on volume #11 inaccessible and destroy any existing files on #11!

   Remember this: **Never copy an entire volume from a small volume to a large volume. Instead, copy each file, one by one, using a wildcard.**

6. The computer now responds with the prompt:

```
Filecopy to what ?
```

This prompt is asking you to specify the *destination* file: the file where you want the copies put. You want to copy each file from the BOOT: volume to unit #11, and you want all of the new copies to have the same names as the originals. Type:

```
#11:$ Return
```

Here, the **$** wildcard is used to give all files copied to the destination volume the same names as the files on the source volume.

7. The copying process now commences, as each file in the BOOT: volume is read into your computer's memory, then copied to the volume assigned to unit #11 on the Winchester disc. The computer displays each file as it is copied. The screen should look something like this:

```
BOOT:SYSTEM_P          ==> V11:SYSTEM_P
BOOT:INITLIB           ==> V11:INITLIB
BOOT:TABLE             ==> V11:TABLE
BOOT:STARTUP           ==> V11:STARTUP
BOOT:SWVOL.CODE        ==> V11:SWVOL.CODE
BOOT:AUTOSTART         ==> V11:AUTOSTART
```

The left side of the display shows the volume ID and the file name of each file copied *from* the source volume BOOT:. The right side of the display shows the volume ID and the file name of each file copied *to* the destination volume on the Winchester disc.

8. When the Filer's command line reappears at the top of the screen, all files have been copied. Remove the BOOT: volume from the flexible disc drive and put it back in its disc box.

Copying the Pascal Subsystems

Now that you have copied the boot files to your Winchester disc, the next step is to copy some of the Pascal subsystems. In this section, you will copy the Editor, Filer and Compiler subsystems to unit #12 of the Winchester disc.

Note

If your Winchester disc has only one volume, copy the subsystems to unit #11.

This procedure assumes you still have the Filer loaded from the last procedure. If you do not, follow the first two steps of "Copying the Boot Files" and come right back.

1. Insert the ACCESS: volume into one of your flexible disc drives. You will copy the Editor and Filer from this volume.
2. Press [F] to invoke the Filer's **Filecopy** command. The computer responds with:

   ```
   Filecopy what file ?
   ```

   In this situation, we want to copy some, but not all, of the files in the ACCESS: volume. The easiest way to do this is to use the ? wildcard, so type:

   ```
   ACCESS:? [Return]
   ```

   The ? wildcard works like the = wildcard, for it allows you to copy several files from the source volume, one at a time. The difference is that ? will prompt you before it copies a file, allowing you to select only those that you want. The = wildcard, if you recall, automatically copies *all* files in the volume, and does not allow you to choose.
3. The computer now asks:

   ```
   Filecopy to what ?
   ```

   Type the destination volume where you want the copies put, and use the $ wildcard to name the new files that same as the originals:

   ```
   #12:$ [Return]
   ```
4. The copying now commences, and the computer asks if you want to copy the first file in the ACCESS: volume:

```
Filecopy ACCESS:FILER ? (Y/N)
```

You will need the Filer for the demonstrations in this part of the manual, so press [Y] to copy it. The file is read from the ACCESS: volume and copied to the volume associated with unit #12. The computer tells you the copy operation is complete with:

```
ACCESS:FILER                    ==> V12:FILER
```

5. The computer now asks if you want to copy the next file in the ACCESS: volume, namely the Editor:

   ```
   Filecopy ACCESS:EDITOR ? (Y/N)
   ```

   Again, press [Y] to copy the Editor. The copies it and then reports:

   ```
   ACCESS:EDITOR                   ==> V12:EDITOR
   ```

6. The remaining files in the ACCESS: volume are of no use right now, so press [N] each time the computer asks if you want to Filecopy them. When the Filer's command line returns to the top of the screen, all files in the ACCESS: volume have been copied or rejected.

7. Next, remove the ACCESS: volume and insert the CMP: volume. Since you will copy only one file from this volume (the Compiler), it is easiest to use the copy procedure described in Chapter 7. Press [F] to invoke **Filecopy** again. When the computer asks:

   ```
   Filecopy what file ?
   ```

   respond with:

   ```
   CMP:COMPILER [Return]
   ```

   When the computer prompts for the destination:

   ```
   Filecopy to what ?
   ```

   respond with:

   ```
   #12:$ [Return]
   ```

8. The computer reports the completion of the copy operation with:

   ```
   CMP:COMPILER                    ==> V12:COMPILER
   ```

Remove the CMP: volume and store it properly.

You have now copied all of the subsystems needed for the demonstrations in the remainder of this book.

Booting From Your Winchester Disc

If your computer has Boot ROM Rev. A, 4.0 or 3.0, your booting procedure is unchanged from the one used in Chapter 2. Reboot your system now. Simply turn your computer on (or press [Reset] if it is already on), and the Boot ROM will find the Pascal system's boot files on your Winchester disc and load them.

After a minute the computer will prompt you to:

```
Please put SYSVOL: in unit #3
  and press the X key...
```

You will remove this prompt later in this chapter when you create a new Autostart file. For now, just press [X] to continue booting. Answer the time and date prompts when they appear.

The Pascal system will also find the Editor, Filer and Compiler on unit #12 and update the **What** command's table accordingly. You will no longer need the flexible discs to boot, or to invoke the Editor, Filer and Compiler.

If your computer has Boot ROM 3.0L, 2.0 or 1.0, you will still need a flexible disc copy of your BOOT: volume. Simply insert the BOOT: volume into your flexible disc drive and turn your computer on (or press [Reset]). The computer must boot the Pascal system from the flexible disc. However, it can still use the Editor, Filer and Compiler stored on unit #12, so you will no longer need the ACCESS: volume.

Note

Flexible disc unit number assignments may change once you begin booting from a Winchester disc. Use the procedure for identifying your flexible discs provided in the "Verifying Unit Number Assignments" section of Chapter 3 to update your System Configuration Table. We recommend you do this now.

Permanently Loading Subsystems

The next step to setting up your environment is to use the **Permanent** command to permanently load frequently-used subsystems into your computer. As demonstrated in Chapter 5, permanent-loading dramatically reduces subsystem access times, thus making program development that much easier.

The number of subsystems you should permanently load depends on how much memory your computer has. Permanent-loading displaces memory that is otherwise available for programs. If memory is scarce and you will be developing large programs, you may want to load only one or two subsystems, or perhaps none at all. Appendix C contains a listing of all Pascal system programs and how much memory each requires. You may want to use this to determine how many subsystems to permanently load.

Which subsystems you should load depends upon how you use the Pascal Workstation. After you use the system for a while, you'll know which programs you use most often and would make good candidates for permanent-loading.

In this example, we will permanently load the Editor and Filer. The Compiler, while used frequently in the demonstrations, requires a great deal of memory. It will just have to be placed on-line whenever it is needed.

To permanently load the Editor and Filer, use this procedure:

1. Return to the Main Command Level. Invoke the **Version** command by pressing [V]. Make a note of the number of bytes of `Total Available Memory` displayed on the screen. This number will decrease as you permanently load programs.
2. Press [Return] twice to return to the Main Command Level. Press [P] to invoke the **Permanent** command.

3. The computer now asks:

   ```
   Load what code file?
   ```

   Make sure the Filer is on-line (either in the ACCESS: volume or in a Winchester disc volume). Type the complete file specification for the Filer, including its volume ID and file name. Don't forget to follow the file name with a period to suppress the automatic appending of the suffix. For example, if the Filer is still on the ACCESS: volume, you would type:

   `ACCESS:FILER.` Return

4. Press V to invoke the **Version** command again, and notice how the number of bytes of total available memory has decreased. The difference has been consumed by the Filer. Press Return twice.

5. Press P again and permanently load the Editor as you did the Filer in step 3. The file name for the Editor is EDITOR.

6. Finally, invoke the **Version** command again and note the amount of memory now available for programming. Press Return twice.

The Editor and Filer are now permanently loaded.

Making a RAM: Volume

The demonstrations to follow will use a RAM: (memory) volume for the system volume. RAM: volumes are very convenient for many applications, but like permanently loaded programs, they do consume memory. Thus, decisions about whether to use a RAM: volume and, if so, how big to make it are governed by the amount of memory in your computer and the size of the programs you'll be developing.

In this example, we use a RAM: system volume because it allows very rapid reading and updating of the workfiles. Since workfiles are used only temporarily, the risks involved with RAM: volumes (e.g., file loss due to power outage, etc.) are minimized.

To create a RAM: volume at unit #50, follow this procedure:

1. At the Main Command Level, press [M] to get the following prompt:

   ```
   *** CREATING A MEMORY VOLUME ***

   What unit number?
   ```

   Type the following line to associate the RAM: volume with unit #50:

   ```
   #50: Return
   ```

2. Next, the computer asks how much memory (in increments of 512-byte blocks) you want to allocate to the RAM: volume:

   ```
   How many 512 byte BLOCKS?
   ```

   We will create a very small RAM: volume of only 20 blocks. Type:

   ```
   20 Return
   ```

3. The computer now asks:

   ```
   How many entries in directory?
   ```

   Since the RAM: volume will only be used to store the workfiles, two directory entries is adequate. However, eight entries is the minimum number you can have, so type:

   ```
   8 Return
   ```

4. The computer now creates the RAM: volume, and reports:

```
#50: (RAM:) zeroed
```

The memory volume has been assigned to unit #50 and has RAM: as its volume ID. To confirm that the memory volume exists, press [F] to enter the Filer, and press [V] to invoke the **Volumes** command. There is now an entry for unit number 50 that looks like:

```
50 # RAM:
```

Press [Q] to return to the Main Command Level.

Setting the System Volume

The next step in setting up your environment is to set the system volume.

One of the main functions of the system volume is to provide the Pascal system with a place to create the temporary files that it needs during compilation and other operations. It is also used to store the temporary workfiles that may be used to speed program development. Thus the system volume must be on-line most all the time, and must have quite a bit of free space where temporary files may be stored.

The system volume is automatically set when you boot the Pascal system. If you have no Winchester disc, the system volume will normally be set to the SYSVOL: volume that you inserted during the booting process. If you do have a Winchester disc, the system volume will usually be assigned to the first volume of the Winchester disc (unit #11). In general, the system volumes is assigned the mass storage device with the fastest access time, to allow temporary files to be written and removed quickly.

In this example, we want the RAM: volume to be the system volume, again because of its extremely fast access time. To change the system volume to RAM:, follow this procedure:

1. At the Main Command Level, press [W] to invoke the **What** command.
2. When the **What** command's table appears on the screen, press [S] to signal that you want to enter a new system volume. The cursor then jumps to the system volume entry, waiting for you to specify the new system volume.

3. Type:

   RAM: [Return]

4. Press [Q] to exit the **What** command.

The RAM: volume is now designated the system volume. Press the space bar (or any other key) to clear the screen.

Some System Volume Considerations

When choosing your system volume, it makes sense to select a volume that can be accessed quickly. Both compilation and movement of workfiles between subsystems will be faster if the system volume is fast.

If performance was the only consideration, the RAM: volume would certainly be the best candidate for the system volume. But reliability is important, and RAM: volumes are vulnerable to power loss. If you use a RAM: system volume for program development and do not occasionally copy the workfile to a disc volume, a power failure would erase all of your efforts.

Thus you have three choices:

1. Use RAM: as the system volume during program development and hope the power doesn't fail.
2. Use RAM: as the system volume and periodically save the workfile on a disc volume. If the power fails, you have only lost changes to the file that were made since you last saved the workfile.
3. Use a disc volume as the system volume. Access time will be slower, but all updates to the workfile are recorded on a disc.

You must decide which option makes the most sense for you. The second option is often a nice compromise between performance and security, but if you have a very fast disc, the last option may be your best bet. If you are developing a short program, or if your computer has powerfail protection, you may choose the first option.

System Date and Time Preservation

If your system **does not** have a battery backed-clock (which is the case for some Series 200 computers), then the only means of preserving the system date and time is the system volume. The Shared Resource Manager (SRM) can be a source for date and time. Other devices only remember the last system **date** written to them. RAM volumes, write-protected devices, and read-only devices cannot be used for this purpose.

Setting the Default Volume

Setting the default volume saves you from typing the volume ID (or unit number) every time you want to retrieve or store a file. Usually, most file activity involves one volume more than the others; it is this most frequently-accessed volume that should be designated the default volume.

If you will accessing one particular disc drive repeatedly, but will be swapping discs in and out of it a lot, it is probably a good idea to set this frequently-used disc drive (unit) as the default "volume." If you do this, any disc that resides in the disc drive automatically becomes the default volume. In this example, we will set flexible disc drive unit #3 as the default volume. Follow this procedure:

1. Press [F] to enter the Filer. Press [P] to invoke the **Prefix-vol** command.
2. The computer responds with:

   ```
   Prefix to what directory ?
   ```

   Make sure unit #3 is empty and type:

   ```
   #3: [Return]
   ```

   The computer responds with:

   ```
   Prefix is #3:
   ```

From now on, *any* flexible disc in unit #3 will be the default volume.

Creating A New Autostart File

At this point, you have set up the programming environment that will be used in the demonstrations to follow. As you become more familiar with the Pascal Workstation, you will find ways to customize this process to suit your particular needs. But whatever you do, you'll probably agree that you would rather not repeat this routine every time you turn your computer on and reboot the Pascal system.

You don't have to. Instead, you can create an **Autostart** file that will do everything for you. It works like this:

After the Pascal system is booted, it searches for a file on the system volume named AUTOSTART. If it finds one, it executes the commands in the file automatically.

The Autostart file is nothing but a "stream" of commands. The system executes this stream in sequence, just as if the commands were typed at the keyboard. Thus, to create an Autostart file to perform most of the tasks this chapter describes, you need only store your keystrokes in a file called AUTOSTART on the system volume that is assigned just after the Pascal system is booted.

There are several things you must keep track of when designing an Autostart file. You must know where you are in the system, where you are going, and how prompts are sequenced. The best way to handle this is to boot the Pascal system, then set up your environment by executing commands from the keyboard, just as you have done in this chapter. As you do this, write down every key you press. This sequence of keystrokes is exactly what you should store in your Autostart file.

Be aware, however, that some actions are difficult to recreate in an Autostart file. For example, actions that require inserting and removing discs at various points in the Autostart process are difficult to implement. For this reason, the default volume will not be set in the following Autostart file, for doing so would require pausing the command stream after the Filer and Editor have been permanently loaded, emptying unit #3, and restarting the command stream again.

Here's how to create an Autostart file that sets up the environment described earlier in the chapter, with the exception of setting the default volume to unit #3:

1. Boot the Pascal system again, using the directions in Chapter 2. This resets the system volume to its original value.

2. Make a note of the system volume setting for future reference. It is given just below the total available memory in the initial display. Enter the date and time.

3. Make sure the volume containing the Editor is on-line, and press [E] to enter the Editor. Press [Return] to create a new file, then press [I] to enter insert mode.

4. Now type the Autostart file exactly as shown. When you see a [Return] key in the following text, press [Return]. **Do not** press [Return] unless explicitly told to do so.

```
=A Remove BOOT: and insert ACCESS:, then press the Return key[Return]
[Return]
[7]
PACCESS:FILER.[Return]
PACCESS:EDITOR.[Return]
M#50:[Return]
20[Return]
8[Return]
WSRAM:[Return]
QV
```

Press the [Select] key to exit insert mode **immediately** after typing `V`. Since all keystrokes are significant in an Autostart file, you should not have any trailing spaces.

When finished, the Autostart file should look like this:

```
=A Remove BOOT: and insert ACCESS:, then press Return

[7]
PACCESS:FILER.
PACCESS:EDITOR.
M#50:
20
8
WSRAM:
QV
```

To illustrate the correlation between the Autostart file and keyboard commands, let's step through the file, line by line:

Line 1 This is a prompt line; it is not executed like a keyboard command but is instead displayed when "streaming" of the Autostart file begins. Prompts are used to make the Autostart file more interactive. In this case, the prompt reminds you when to remove the BOOT: volume and insert the ACCESS: volume so that the Filer and Editor can be permanently loaded. (Useful when booting from floppy disc.) Execution of the commands in the file is suspended until Return is pressed. For more information on Autostart prompts, refer to the **Stream** command section in the *Pascal Workstation System Manual.*

Lines 2 Since the commands in the Autostart file are executed immediately after the Pascal system is booted, the first command in the file must respond to the first prompt that appears after booting. Remember that the first prompt that appears after booting is the date prompt. The Return key on this line is a response to this prompt, accepting the existing date (see the section "Setting the Date and Time" in the chapter "A Few Preliminaries" found in this manual).

Line 3 The next prompt is the time prompt. The `[7]` does not change the existing time but sets the time zone to +7 hours and puts the system at the Main Command Level. See the beginning of the chapter "A Few Preliminaries" in this manual for example time zones and how to set the date and time.

Line 4 This line invokes the Main Command Level's **Permanent** command to permanently load the Filer from the ACCESS: volume. `P` invokes the command, and `ACCESS:FILER`. Return is the response given to the **Permanent** command's prompt:

```
Load what code file?
```

Line 5 This line invokes the **Permanent** command to permanently load the Editor from the ACCESS: volume.

Line 6 This line invokes the **Memvol** command to create a RAM: volume. `M` invokes the command. `#50`Return is the response given to the first prompt in the **Memvol** command:

```
What unit number?
```

Line 7 The system is still in the middle of making a RAM: volume. This line, 20[Return], is the response given to the second prompt in the **Memvol** command:

```
How many 512 byte BLOCKS?
```

Line 8 This line, 8[Return], is the response given to the last prompt in the **Memvol** command:

```
How many entries in directory?
```

Line 9 This line invokes the **What** command to set the system volume to the RAM: volume just created. `W` invokes the command. `S` indicates to the **What** command that you want to change the entry for the system volume. `RAM:`[Return] enters RAM: as the new system volume.

Line 10 This line quits the **What** command and invokes the **Version** command. `Q` quits the **What** command. `V` invokes the **Version** command, allowing you to enter the correct values for the time and date.

Modifying the Autostart File for a Winchester Disc

If you have stored the Filer, Editor, and BOOT: (or BOOT2:) files on a Winchester disc, you must change a few things in the Autostart file. Skip to the next section if you have no Winchester disc.

Since the Filer and Editor are stored on a Winchester disc volume, you don't need the prompt line to remind you to insert the ACCESS: volume. You must also substitute the volume ID of Winchester volume containing the Filer and Editor for the ACCESS: volume in lines 4 and 5. Change your Autostart file like this:

1. Position the cursor under the "=" on line 1. Press [D] to enter delete mode, and press [Return] to delete line 1. Press [Select] to exit delete mode.
2. Position the cursor under the "A" in `ACCESS:` in line 4. Press [D] to enter delete mode, and hold down the space bar until `ACCESS` is deleted (do not delete the :). Press [Select] to exit delete mode. Press [I] to enter insert mode and type the volume ID of the Winchester disc volume containing the Filer. For example, if the Filer is stored on unit #12, and you have not changed the volume ID assigned to this volume by MEDIAINIT, you would type `V12` (or `#12`). Press [Select] to exit insert mode.

3. Position the cursor under the "A" in **ACCESS** on line 5. Press [D] to enter delete mode, and hold down the space bar until **ACCESS** is deleted (do not delete the :). Press [Select] to exit delete mode. Press [I] to enter insert mode and type the volume ID of the Winchester disc volume containing the Filer. For example, if the Filer is stored on unit #12, and you have not changed the volume ID assigned to this volume by MEDIAINIT, you would type **V12** (or **#12**). Press [Select] to exit insert mode.

Your Autostart file should look something like this, assuming that the Editor and Filer are on V12: of your Winchester disc (Note: there is a blank line at the top of the file):

```
[7]
PV12:FILER.
PV12:EDITOR.
M#50:
20
8
WSRAM:
QV
```

Storing an Autostart File

The Autostart file must be stored on the system volume that is assigned after booting, in a file named AUTOSTART. If you have not changed the system volume setting since you booted the Pascal system, use this procedure to save the Autostart file. If you *have* changed the system volume, either change it back to its original setting with the **Newsysvol** command, or reboot the Pascal system.

1. Press [Q] to exit the Editor.
2. When the editor presents your exiting options, press [W] to choose **Write to a file name and return.**
3. Make sure the original system volume is on-line. In particular, if the flexible disc volume SYSVOL: (or some other flexible disc volume) was assigned the system volume after booting, make sure this volume is inserted into one of your flexible disc drives.

4. The computer now asks:

```
Name of output file (<ret> to return) -->
```

Type the following line to store the Autostart file on the system volume in a file named AUTOSTART (don't forget the period after the file name):

```
*AUTOSTART. Return
```

The * is a symbol for the system volume.

Note

If an Autostart file already exists on the system volume, you may get a message asking if you want to rewrite it, overwrite it, purge it, or none of the above. Press R to rewrite it.

---

5. After the file is written, the following prompt will appear:

```
Exit from or Return to the editor ?
```

Press E to exit.

Booting with an Autostart File

To boot the Pascal system, simply use the procedure you learned in Chapter 2. The Autostart file will be found and "streamed." If something goes wrong, make sure you have stored the file using the file name AUTOSTART, on the system volume that is assigned after booting. Also check that the file is designed correctly, as described in the next section.

After booting the Pascal system, there is one "loose end" that the Autostart file didn't perform for you. The default volume must be set manually. To do this:

1. Answer the date and time prompts.
2. Press F to enter the Filer, and press P to get the following prompt:

```
Prefix to what directory ?
```

3. Make sure that unit #3 is empty, and type the following response:

```
#3: Return
```

4. The computer reports:

```
Prefix is #3:
```

Press [Q] to exit the Filer.

Confirming an Autostart File

If you would like to confirm that an Autostart file is doing what it's supposed to do, try a dry run from the keyboard. Just boot the Pascal system, and type all keystrokes in your Autostart file directly from the keyboard, ignoring prompts and prompt calls.

A Program Development Session 11

Now that a comfortable environment is established, it's time to design, debug and execute a simple Pascal program. This chapter demonstrates how the components of the Pascal system work together to make program development easier.

In keeping with the spirit of Murphy's law, most every task we describe in this chapter is first performed incorrectly. We think it best to expose you to many of these common errors now, rather showing you a flawless example and leaving it to you to recover from errors when you encounter them on your own.

In this chapter, you will:

- Create a data file
- Create a simple program
- Respond to system, syntax and run-time errors that arise
- Run the compiled program
- Back up the program

Getting Ready for Chapter 11

If you have not already done it, set up the programming environment as described in Chapter 10. Don't forget to set the default "volume" to unit #3 (with disc removed); the Autostart file will not do this for you.

You will need one empty, initialized disc for storing the program, data and listing files you will create in this demonstration. Insert this disc into unit #3 to make it the default volume.

Also, have the disc labeled CMP: handy, unless you have copied the Compiler to a Winchester disc. The Compiler must be on-line before you compile a program.

Creating the Data File

In this chapter, you will create a program that reads a list of numbers from a data file and prints them on the screen. Begin by creating the data file.

The data file is nothing more than a list of integers. To create it, make sure you're at the Main Command Level and press [E] to enter the Editor. Press [Return] to create a new file. (If the Editor automatically reads in a workfile, use the Filer's **New** command to clear it.) Press [I] to enter insert mode, and type in the following list of numbers exactly as shown:

```
1
2
3
4
5
6
7
8
9
10
```

Press [Select] to exit insert mode. Press [Q] to quit the Editor and look over your exiting options.

You must decide whether using a workfile offers any advantage in this case. Since you will do nothing more with this data right now, the simplest exiting option is to press [W] to write the data directly to a file. While you could press [U] to update the workfile, then enter the Filer, use the **Save** command to store the workfile, and use **New** to clear the workfile, why take all of these steps when you can do it in one?

The program will expect to find its data in a file named DATA.TEXT in the default volume. To store the data there, press [W] to get the following prompt:

```
Name of output file (<ret> to return) -->
```

Make sure your new, initialized disc is in unit #3 and type the following response to store the data in the file DATA.TEXT in the default volume:

```
DATA [Return]
```

The computer confirms the save operation with something like:

```
Your file is 40 bytes long.
Exit from or Return to the editor?
```

Press [E] to exit from the Editor.

Creating the Program

Next, you will create the program that reads and prints the data file.

Press [E] to enter the Editor, and press [Return] to create a new file. Press [I] to enter insert mode and type in the following Pascal program exactly as shown. For those of you familiar with Pascal, this program contains intentional errors, so don't try to correct them.

```
PROGRAM COUNT(INPUT,OUTPUT)
VAR NUM:INTEGER;
    INFILE:TEXT;
  BEGIN
    RESET(INFILE,'DATA.TEXT');
    WHILE NOT EOF(INFILE) DO
      BEGIN
        READ(INFILE,NUM);
        WRITELN(NUM)
      END
  END.
```

Press [Select] to exit insert mode, then press [Q] to quit the Editor.

Again, you must decide whether to use the workfile. This is a program, and programs often contain errors. You will probably compile the program, find an error, edit the program to correct the error, compile the program again, find another error, etc. Since you'll be moving this file from one subsystem to another during this process of step-wise refinement, a workfile would be a definite asset.

Press [U] to update the workfile. The program you have just created is now stored in the file WORK.TEXT in the system volume.

Because the system volume is RAM:, there is always the risk of losing the program due to a power loss. To guard against this, enter the Filer and press [S] to save the contents of the workfile into a file in your default volume. The computer prompts:

```
Save as what file?
```

Respond by typing:

```
COUNT [Return]
```

to store the program in a file named COUNT.TEXT in the default volume (the suffix .TEXT is automatically appended). Now if there is a power loss, you can at least recover the original version of the program. As you make changes to the workfile, it's a good idea to periodically update the permanent disc file with the most current version.

Exit the Filer.

Compiling the Program

Before you can run the program, you must compile it. Find the disc labeled CMP: and insert it into one of your flexible disc drives. Press [C] to run the Compiler, and wait for the following display:

```
Pascal [Rev 3.2  1/15/87]   15-Feb-87  08:41:07

     Copyright 1987 Hewlett-Packard Company.
               All rights reserved.

Printer listing (l/y/n/e)?
```

If you inserted the CMP: disc into unit #3, remove it and reinsert the disc containing the file COUNT.TEXT.

The workfile is now starting to pay off. Because a workfile exists, the Compiler assumes it contains the program you want to compile, and doesn't ask you for a file name. This saves you from having to type the file name all over again. The Compiler proceeds directly to the prompt:

```
Printer listing (l/y/n/e)?
```

Press [N] to skip the listing. The program is then compiled, and the following error message appears on the screen:

```
PROGRAM <<<<
Line 1, error 901
<sp>=continue, <esc>=terminate, E=edit
```

Note

If you created a larger RAM: volume when designing your Autostart file in Chapter 10, this message may not appear. If it doesn't skip to the section called "A Syntax Error."

This message says that the error was detected in line 1 of the program, and that the error number is 901. By itself, this message doesn't say a lot. Get out the *Pascal Workstation System Manual* and lift the tab labeled **Error Messages**. Find the section called "Pascal Compiler Syntax Errors" and look up error 901 toward the very end of this section. Error 901 says: "insufficient space to open ref file."

If you were to read the Compiler chapter of the *Pascal Workstation System Manual*, you would discover that a ref file is one of the temporary files the Compiler creates while compiling a program. Since temporary files are stored on the system volume, RAM:, this message tells you that the system volume contains insufficient free space for writing temporary files. To correct this problem, the RAM: volume must be made larger.

While you might be inclined to use the **Memvol** command to make a new, larger RAM: volume at unit #50, the procedure is not quite so simple. First of all, because the RAM: volume is created in the Autostart file, the Autostart file must be modified to make any changes to the RAM: volume permanent. Otherwise, the Autostart file will create the same small RAM: volume every time you boot. Secondly, when you create a second RAM: volume at the same unit number as the first, the memory allocated to the first RAM: volume is not recovered. Thus, changing the size of a RAM: volume in this way is a waste of memory.

To change the size of the RAM: volume, you must edit the Autostart file and reboot the Pascal system. While you could do this now, a better plan is to change the system volume from RAM: to the volume in unit #3, finish the program development session, and change the Autostart file later. You will lose the rapid access time that a RAM: system volume provides, but since this example deals with such a small program, it won't slow you down too much.

Changing the System Volume

To change the system volume from RAM: to the flexible disc volume in unit #3, use the Main Command Level's **Newsysvol** command.

The bottom line of the display tells you to press the space bar to continue, so press it now. Then press [N] to invoke **Newsysvol**. The following prompt will appear:

```
What new system unit number?
```

Type:

```
#3: [Return]
```

When the Main Command Level's command line returns to the top of the screen, the new system volume has been set.

The volume in unit #3 is a reasonable choice for a system volume, for it does contain plenty of free space. Flexible discs, however, are rather slow to access, so if you have a free Winchester disc volume available, you may want to set the system volume there.

Compiling the Program Again

Since the last error was caused by your programming environment and had nothing to do with the syntax of your program, there is no need to edit the program. Insert the CMP: volume into one of your disc drives and press [C] to proceed directly to the Compiler.

If you inserted the CMP: disc into unit #3, wait for the following prompt to appear, then remove CMP: from the disc drive:

```
Compile what text?
```

Reinsert the disc containing the COUNT program into unit #3.

Now why did this prompt appear? You do, after all, have a workfile. But remember, you changed the system volume to the volume in unit #3. The workfile is still in the *old* system volume, RAM:. The computer searches the system volume in unit #3 for WORK.TEXT, and doesn't find it. Because there is no workfile, the Compiler doesn't know which program to compile, and asks for the name and location of the file. Since you're already in the middle of compiling, give the Compiler the file COUNT.TEXT that you saved in the default volume—you can create a new workfile at the next convenient opportunity.

Type the following response to compile the COUNT program:

```
COUNT [Return]
```

The suffix, .TEXT, is assumed. The computer then lets you select a printer listing option:

```
Printer listing (l/y/n/e)?
```

Press [L] to send the listing to a file. The computer then asks for the name of the file:

```
What listing file?
```

Type the following response to store the listing in the file named COUNTLIST.TEXT on the default volume:

```
COUNTLIST[25] [Return]
```

The [25] tells the computer how much space to allocate for the listing file (in blocks). This file size specification is required. One listing page will occupy approximately 25 blocks. Since this listing will not exceed one page, 25 blocks is plenty.

If you had a workfile, the output file would automatically be WORK.CODE. But since you don't, the Compiler prompts for the name of the output file where the compiled code should be stored:

```
Output file (default is "#3:COUNT.CODE") ?
```

The Compiler suggests an output file name of COUNT.CODE, stored in the default volume. This is a logical choice, so press [Return] to accept it.

A Syntax Error

The Compiler begins compiling the program and displays the following messages:

```
<    0>
PROGRAM COUNT(INPUT,OUTPUT)
  VAR <<<<
Line 2, error 14

COUNT             [593436]
<    4>.
No codefile generated.

11 lines.  1 error.  No warnings.
```

This tells you that another error was detected. This time, it is error number 14, and it was found in line 2 of the program. Because an error was detected, no output (code) file was created.

Before you investigate the error, use the Editor to look at the listing file that was generated. Press [E] to enter the Editor. Since no workfile exists, the Editor asks for the name of the file you want to edit:

```
File? (<ret> for new file, <stop> exits)
:
```

Type the following response to retrieve the listing file, COUNTLIST.TEXT, from the default volume:

```
COUNTLIST [Return]
```

The file is then displayed:

```
Pascal [Rev 3.2  1/15/87] COUNT.TEXT            15-Feb-87 10:43:02 Page 1

     1:D        0 PROGRAM COUNT(INPUT,OUTPUT)
     2:D    -4  1 VAR NUM:INTEGER;
                  ^
>>>>>> Error at COUNT.TEXT/2:  14
     3:D  -668  1     INFILE:TEXT;
     4:C        1   BEGIN
     5:C        1     RESET(INFILE,'DATA.TEXT');
     6:C        1     WHILE NOT EOF(INFILE) DO
     7:C        2       BEGIN
     8:C        2         READLN(INFILE,NUM);
     9:C        2         WRITELN(NUM)
    10:C        2       END
    11:C        2   END.

1 error. See line 2. No warnings.
```

Notice the error message appearing immediately below line 2:

```
>>>>>> Error at COUNT.TEXT/2:  14
```

This message tells you that an error was detected in the file COUNT.TEXT at line 2, and that the number of the error is 14. A caret points to the location in line 2 where the error was detected (i.e., to the VAR declaration).

Look up error 14 in the "Pascal Compiler Syntax Errors" section of the *Pascal Workstation System Manual.* This error number decodes to: "Expected a semicolon ";"." Apparently, the program is missing a semicolon in line 2.

Line 2 of the program ends with a semicolon as it should, but line 1 does not. The Compiler could not detect the error on line 1 until after it began reading line 2, which explains why the error was attributed to line 2.

Press [Q] to quit the Editor, followed by [E] to exit without updating.

Correcting the Syntax Error

Now press [E] to reenter the Editor, and type the following response to edit your program:

```
COUNT [Return]
```

The program is loaded and made available for editing.

To correct the error, hold down the space bar and position the cursor to the end of line 1. Press [I] to enter insert mode, and type the following to add a semicolon and exit insert mode:

```
; [Select]
```

When finished, the first line should look like this:

```
PROGRAM COUNT(INPUT,OUTPUT);
```

Press [Q] to exit the Editor. To avoid typing the file name COUNT each time you enter a new subsystem, press [U] to create a workfile in the new system volume. From now on, when you enter a new subsystem, the program will be loaded automatically.

The Third Compilation

Insert the CMP: volume into one of your disc drives and press [C] to run the Compiler. Wait for the following prompt to appear:

```
Printer listing (l/y/n/e)?
```

Note

If you inserted the CMP: volume into unit #3, the following prompt will appear: `Mount *WORK.TEXT and press <space>`

The computer is asking for the system volume, containing the workfile, that you removed from unit #3. Replace the CMP: volume with the system volume containing the workfile, and press the space bar.

Press [L] to get a fresh listing file of the corrected program. When the computer asks:

```
What listing file?
```

type the following response again to overwrite the original listing:

```
COUNTLIST[25] [Return]
```

Compilation then gets underway, and the following messages are displayed:

```
<    0>.
COUNT             [593436]
<    4>.

11 lines,  No errors.  No warnings.
```

No errors were detected; the code file WORK.CODE was created and stored in the system volume in unit #3. Press the space bar to clear the screen.

Running the Program

To run the program, press [R] at the Main Command Level to invoke the **Run** command. **Run** loads, compiles (if necessary), and then runs the workfile.

The following display appears on the screen:

```
Restart with debugger ?
         1
         2
         3
         4
         5
         6
         7
         8
         9
        10

------------------------------------------------
error -10: tried to read or write past eof
PC value:    -381574
```

A Run-time Error

The numbers 1 through 10 in this display are the output from the program. But at the end of program execution, an error number -10 was reported. If you look on the first page of the Error Messages section of the *Pascal Workstation System Manual*, you'll find that an error -10 indicates an I/O related error. The Pascal system has printed out a description of the error, namely that the program tried to read past the end of the data file.

This problem can be corrected by changing the READ statement in line 8 to a READLN statement. To do this, press [N] (No) to respond to the debugger prompt at the top of the screen, and press [E] to go back to the Editor.

Correcting the Run-time Error

The workfile is automatically read into the Editor. Press [Return] to move the cursor to the beginning of the line:

```
READ(INFILE,NUM);
```

Press the space bar to position the cursor under the left parentheses (. Press [I] to enter insert mode, and type:

```
LN [Select]
```

The line should now look like this:

```
READLN(INFILE,NUM);
```

Press [Q] to quit the Editor, and then [U] to update the workfile and leave.

The Final Compilation

Make sure the Compiler is on-line, then press [C] to load it. If you have only one flexible disc drive and must remove the system volume in order to load the Compiler, be sure to reinsert the system volume as you did earlier.

Again, because a workfile exists, you need only answer the printer listing prompts; the Compiler's input and output files are assumed to be the workfiles WORK.TEXT and WORK.CODE. When the prompt appears, press [L], then type the following line to send the listing to the file COUNTLIST.TEXT in the default volume:

```
COUNTLIST[25] [Return]
```

Compilation proceeds without incident, and the code file is generated. The display looks like this:

```
<    0>.
COUNT              [593436]
<    4>.

11 lines,  No errors.  No warnings.
```

Running the Program Again

Press [R] to load and run WORK.CODE, which contains the latest code version of program COUNT. The program runs successfully, and displays its output on the screen:

```
1
2
3
4
5
6
7
8
9
10
```

To rerun the program, press [U] to invoke the **User** restart command.

Saving the Finished Program

As it stands, the only copies of the finished program are stored in the workfiles. WORK.TEXT contains the uncompiled (text) version, and WORK.CODE contains the compiled (code) version. Since the workfiles are only temporary files, and now that program development is completed, you should save their contents into more permanent files.

Enter the Filer, and press [S] to invoke the **Save** command. The computer will ask:

```
Save as what file ?
```

Type the following response:

```
COUNT [Return]
```

Since the file COUNT.TEXT already exists on the default volume, the computer wants to know what to do with the old file before it saves the new one. It reports something like:

```
V3:COUNT.TEXT
exists ... Remove/Overwrite/Neither ? (R/O/N)
```

Press [R] to remove the old file and save the new corrected version. When finished, the computer displays:

```
V3:WORK.TEXT                    ==> V3:COUNT.TEXT
```

A nice feature of the **Save** command is that it will save *both* workfiles if they exist. Since WORK.CODE is also present in the system volume, **Save** copies it to the file COUNT.CODE in the default volume. COUNT.CODE does not already exist in the default volume, so the computer saves it immediately, rather than first prompting you as it did with COUNT.TEXT.

The computer reports:

```
V3:WORK.CODE                    ==> V3:COUNT.CODE
```

And then:

```
Source file saved & Code file saved
```

Once the workfiles are saved, you can clear the workfiles by pressing [N] (**New**). Do this now, and WORK.TEXT and WORK.CODE are removed from the volume in unit #3 (the system volume).

Exit the Filer, and press the space bar to clear the screen.

Backing Up Your Program

The last step of program development is to make a back-up copy of your completed program.

To make a back-up copy of the COUNT program, get out another initialized flexible disc and label it as your back-up disc. Follow one of the procedures below, depending on whether you have one or two flexible disc drives.

If You Have Two Flexible Disc Drives...

1. Leave the disc containing program COUNT into the unit #3, making it the default volume.
2. Insert the back-up disc into your other flexible disc drive (in this example, we assume it is unit #4).
3. Enter the Filer, and press [F] to invoke the **Filecopy** command. The following prompt will appear:

   ```
   Filecopy what file?
   ```

4. Type the following response to indicate you want to copy the file COUNT.TEXT from the default volume.

   ```
   COUNT.TEXT [Return]
   ```

Note

There is no reason to back up COUNT.CODE, for you can always recompile COUNT.TEXT to produce it. Making a *back-up* copy of .CODE files is generally not necessary, but you will want to keep a *working* copy.

Notice that you must give both the file name and the suffix when specifying a file to the **Filecopy** command. **Filecopy** does not automatically append the suffix .TEXT as do some of the other commands; it uses the name exactly as typed.

5. The computer then asks:

   ```
   Filecopy to what?
   ```

   If your back-up disc is in unit #4, type:

   #4:$ Return

   Otherwise, substitute the unit number of the drive containing the back-up disc for #4: above. The **$** wildcard is used to give the back-up copy the same name as the original.

The file COUNT.TEXT is now copied to the back-up disc. When the Filer's command line reappears at the top of the screen, remove the back-up disc and store it in a safe place, away from the original copy of the program.

Exit from the Filer, and skip to the section, "Changing the Autostart File."

If You Have One Flexible Disc Drive...

1. Leave the disc containing program COUNT into the unit #3, making it the default volume.
2. Enter the Filer, and press [F] to invoke the **Filecopy** command. The following prompt will appear:

   ```
   Filecopy what file?
   ```

3. Type the following response to indicate you want to copy the file COUNT.TEXT from the default volume.

   `COUNT.TEXT` [Return]

Note

There is no reason to back up COUNT.CODE, for you can always recompile COUNT.TEXT to produce it. Making a *back-up* copy of .CODE files is generally not necessary, but you will want to keep a *working* copy.

Notice that you must give both the file name and the suffix when specifying a file to the **Filecopy** command. **Filecopy** does not automatically append the suffix .TEXT as do some of the other commands; it uses the name exactly as typed.

4. The computer then asks:

   ```
   Filecopy to what?
   ```

   Since you have only one flexible disc drive, type the following response:

   `#3:$` [Return]

   The computer will prompt you at the appropriate time to exchange the disc containing COUNT for the back-up disc. The **$** wildcard is used to give the back-up copy the same name as the original.

5. The computer reads COUNT.TEXT from the disc into memory, then prompts you to insert the back-up disc with:

```
Please mount DESTINATION in unit #3
'C' continues, <esc> aborts
```

   Remove the disc containing COUNT.TEXT, and insert the back-up disc. Press [C] to proceed with the copy operation.

The file COUNT.TEXT is now copied to the back-up disc. When the Filer's command line reappears at the top of the screen, remove the back-up disc and store it in a safe place, away from the original copy of the program.

Exit from the Filer.

Changing the Autostart File

The last thing to do is create a larger RAM: volume in your Autostart file, one that is of more practical use. Recall that the present size causes errors during compilation.

Press [E] to enter the Editor. When asked for the file to be edited, type the volume ID (or unit number) of the original system volume, followed by **AUTOSTART**. (don't forget the period). Make sure the original system volume is on-line, then press [Return]. For example, if the system volume after booting the Pascal system was the SYSVOL: volume, you would insert the disc labeled SYSVOL: into one of your disc drives, and type the following line in response to the prompt:

```
SYSVOL:AUTOSTART. [Return]
```

Change the RAM: volume size from 20 blocks to a larger size with the **eXchange** command. To do this, press [Return] to position the cursor to the line containing 20 (see the following illustration).

```
=A Remove BOOT: and insert ACCESS:, then press Return

[7]
PACCESS:FILER.
PACCESS:EDITOR.
M#50:
20
8
WSRAM:
QV
```

Now press [X], type the new size and press [Select]. In this example we have changed the size to 60 blocks. If you choose a size of 100 blocks or greater, use insert mode to add the last digit. Your file should now look something like this:

```
=A Remove BOOT: and insert ACCESS:, then press Return

[7]
PACCESS:FILER.
PACCESS:EDITOR.
M#50:
60
8
WSRAM:
QV
```

Now, press [Q] to exit the Editor, and press [S] to store the new Autostart file back into the original system volume.

Press [E] to exit the Editor, and reboot the Pascal system according to directions in Chapter 2. The Autostart file will be streamed, and the new, larger RAM: volume will be established.

One last point: If you find that you constantly need to change the size of the RAM: volume for your various applications, it may be best to leave RAM: volume creation out of the Autostart file. This you can decide when you become more familiar with the system.

Index to Tasks

Task Reference A

General Instructions

If you are using this reference to get started with the Pascal system, read through the procedures in the order they are presented. Occasionally, you may have to skip ahead to a procedure, depending upon how you will be using the system.

To find the procedure for a particular task, refer to the index on the tab. If you have difficulty understanding how to do something, check the **Additional Information** heading and turn to one of the page references listed there.

Some blank reference pages are provided at the end of this appendix. Use them to document your own procedures.

A Typical Reference Page

Each reference page contains the following information:

Volumes on-line:	Lists all volumes that *may* be required by the procedure. If you have permanently loaded some subsystems or copied them to a Winchester disc, you may not need all of the volumes listed here.
Subsystem:	Gives the name of the subsystem that contains the command used in the procedure.
Command:	Gives the primary command used in the procedure.
Description	Provides a brief description of the procedure.
Procedure	Gives step-by-step instructions for the task.
Additional Information	Provides page references for further clarification of the task and related information.

Booting

Volumes on-line: BOOT:, or BOOT2:, SYSVOL:
Subsystem: None
Command: None

Description

Booting loads the kernel of the Pascal system from the boot volume into RAM, and turns control over to the Command Interpreter (the Main Command Level).

Procedure

1. Turn off the computer. Turn on all external disc drives.
2. Insert the boot volume into unit #3 (Use BOOT2: if you have a Series 300 computer with bit-mapped display, use BOOT: if you have a Series 200 computer or are using the HP 98546A display compatibility interface in your Series 300 computer.)
3. If you have a second flexible disc drive, insert the SYSVOL: volume there.
4. Turn your computer on.
5. If you have only one flexible disc drive, the following prompt may appear:

   ```
   Please put SYSVOL in unit #3
      and press the X key
   ```

   If it does, remove the boot volume, insert the SYSVOL: volume, and press [X].
6. When the following prompt appears at the top of the screen, the Pascal system is booted and ready for use:

   ```
   New system date ?
   ```

   Remove the boot and SYSVOL: volumes and store them properly.

Additional Information

Booting: 18
Booting errors: 20
Flexible disc handling: 10

Setting the Date and Time

Volumes on-line: None
Subsystem: Main Command Level
Command: **Version**

Description

System date and time prompts are displayed after the Pascal system is booted. They can be re-displayed at any time by invoking the **Version** command.

Procedure

1. Look for the following prompt at the top of the screen:

   ```
   New system date ?
   ```

   If the date prompt does not appear, press [V] to invoke the **Version** command.

2. If the date is correct, press [Return]. Otherwise, type the date in the form **dd-mmm-yy** where:

   - **dd** is one or two numbers representing the day
   - **mmm** is the first three letters of the month
   - **yy** is the last two digits of the year

3. Press [Return]. The following prompt will appear:

   ```
   New system clock time [zone] ?
   ```

   Type the time in the form **hh-mm-ss [tz]** where

   - **hh** is one or two numbers representing the hour on a 24-hour clock
   - **mm** is one or two numbers representing the number of minutes past the hour
   - **ss** is one or two numbers representing the number of seconds past the minute (ignore if you wish - it will set to 00)
   - **tz** is a positive or negative number indicating the time zone corrction value. See table, next page.

4. Press [Return] to return to the Main Command Level.

You can choose to leave off the right-most item or items when specifying the time. For example, you can specify just hours or specify just hours and minutes, leaving off the seconds. Since the time zone is delimited by the brackets, you can specify it independently of the time.

The time zone value is the number of hours your local clock would have to be moved ahead (positive) or back (negative) to reach Greenwich Mean Time. Enter `[0]` if you cannot determine your time zone.

Standard Time Zones

Time Zone	City	Value
	Honolulu	10
Pacific	San Francisco	8
Mountain	Denver	7
Central	Chicago	6
Eastern	New York	5
	London	0
	Paris	-1
	Cairo	-2
	Hongkong	-8
	Tokyo	-9
	Sydney	-10

Note that when you set the system volume and the system date is `1 JAN 70` , the system date will be set by using the date of the new system volume.

Additional Information

Time Zones: 23
Entering the date: 25
Entering the time: 26
Date and Time Preservation
Version command: 122
Date and Time Preservation: 27

Loading Subsystems

Volumes on-line: See procedure below
Subsystem: Main Command Level
Command: See procedure below

Description

The Pascal system is composed of the Editor, Filer, Compiler, Assembler and Librarian subsystems. Each subsystem is entered by pressing the upper-case letter in the subsystem name as it appears in the Main Command Level's command lines. The command lines run across the top of the screen and look like this

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

Press [?] to get the second command line:

```
Command: Assembler Debugger Memvol Newsysvol Permanent Stream User What ?
```

The appropriate volume must be on-line when you load a subsystem (see the following procedure).

Procedure

To enter a subsystem from the Main Command Level, use the table below. Make sure the volume (disc) in the **Volume** column is on-line and press the key in the **Key** column.

Subsystem	Volume	Key
Compiler	CMP:	C
Editor[1]	ACCESS:	E
Filer	ACCESS:	F
Librarian	ACCESS:	L
Assembler	ASM:	A

See the procedure, "Exiting From Subsystems" for exiting instructions. For the moment, you may press Stop to exit, but heed the cautions in Chapter 1.

Additional Information

Main Command Level: 4, 114
Compiler: 161, 194, 200, 203, 264
Editor: 58, 114, 127
Filer: 60, 114, 143
Librarian: 164
Assembler: 164

[1] After pressing E, press Return to create a new file, or type the file specification of the file you want to edit and then press Return.

Exiting From Subsystems

Volumes on-line: any
Subsystems: any
Command: **Quit**

Description

The Editor, Filer and Librarian subsystems are exited by pressing [Q], which corresponds to the **Quit** command in the command line for each of these subsystems. Each of these subsystems can be exited only while at the top-most level of the subsystem; i.e., the command line must be displayed at the top of the screen.

The Editor may present several exiting options after you press [Q]. See the "Storing a File (Editor)" procedure for details.

The Compiler and Assembler are exited automatically after compilation or assembly is complete. Execution of these subsystems may be stopped by aborting one of their prompts (see the "Responding to Prompts" procedure).

All subsystems may be exited at any time by pressing [Stop]. This should be used as a last resort, however. More graceful means are preferred.

Additional Information

Compiler: 161, 194, 200, 203, 264
Editor: 58, 114, 127
Filer: 60, 114, 143
Librarian: 164
Assembler: 164

Invoking Commands

Volumes on-line: ACCESS:
Subsystems: Main Command Level, Editor, Filer, Librarian
Commands: any

Description

Most subsystems in the Pascal Workstation contain a set of commands. The active commands are listed in the command line at the top of the screen (pressing [?] displays a second command line). Commands are invoked by pressing the upper-case letter in the command name as it appears in the command line.

Procedure

1. Enter the desired subsystem (see the "Loading Subsystems" procedure).
2. Find the command in the command line that will perform the desired task.
3. Press the key corresponding to the upper-case letter in the command name to invoke the command.

Additional Information

Command lines: 40
Invoking commands: 40

Exiting from Commands

Volumes on-line: any
Subsystems: any
Commands: any

Description

Some commands (primarily in the Editor) provide operating instructions at the top of the screen. Such instructions usually tell which key(s) to press to exit the command. Keys are represented by abbreviated labels enclosed in angle brackets < >. The key designations you see on the screen depend upon which keyboard you have. Common key designations include:

For the HP 46020A and 46021A Keyboards:

- `<sel>` - press [Select]
- `<esc>` - press [ESC]
- `<ret>` - press [Return]

For the HP 98203A Keyboard:

- `<exc>` - press [EXEC]
- `<sh-exc>` - hold down [SHIFT] and press [EXEC]
- `<ent>` - press [ENTER]

For the HP 98203B and 98203C Keyboard:

- `<exc>` - press [EXECUTE]
- `<sh-exc>` - hold down [SHIFT] and press [EXECUTE]
- `<ent>` - press [ENTER]

Procedure

1. To exit from a *command*, press the key(s) indicated in the command line.
2. If no clues are given in the command line, try pressing:

- [ESC] if you have an HP 46020A or 46021A keyboard
- [SHIFT] [EXECUTE] if you have an HP 98203B/C keyboard
- [SHIFT] [EXEC] if you have an HP 98203A keyboard

To exit from (abort) a *prompt*, see the "Responding to Prompts" procedure.

Additional Information

Responding to prompts: 44
Editor command lines: 128
Command lines: 40

Responding to Prompts

Volumes on-line: any
Subsystems: any
Commands: any

Description

After a command is invoked, it may present a series of prompts that you must answer before the operation is performed. There are several different types of prompts, classified by the response that is expected. Each type is described below:

Straight Questions

To respond, type your answer, and press [Return]. Abort the prompt by pressing [Return] without an answer.

Example: `Volume ID ?`

Yes/No Prompts

To respond, press [Y] for yes or [N] for no. Pressing [N] aborts the prompt.

Example: `Are you SURE you want to proceed ? (Y/N)`

Value Range Prompts

To respond, type a value that falls within the specified range, then press [Return]. The range of acceptable values is enclosed within square brackets `[ ]`. Abort the prompt by pressing [Return] *unless* it also has a default value (as does the one below), in which case press [Stop].

Example: `Interleave factor? [1..15] (defaults to 1)`

Prompts with Default Values

To respond, press [Return] to accept the default value in parentheses, or type your own response and press [Return]. Abort the prompt by pressing [Stop].

Example: `Number of directory entries (80) ?`

Additional Information

Responding to prompts: 44

Initializing (Formatting) Flexible Discs

Volumes on-line: ACCESS:
Subsystem: Main Command Level
Command: **eXecute**

Description

New flexible discs must be initialized (formatted) before they can be used. Initialization is performed by a program stored in the file MEDIAINIT.CODE in the ACCESS: volume. The program is run by invoking the **eXecute** command from the Main Command Level.

CAUTION

Only new discs should be initialized. If you want to re-initialize a disc for any reason, be sure to save its contents on another disc.

Procedure

1. Insert the ACCESS: volume into the same drive where you inserted the BOOT: volume when booting. Press [X] to invoke the **eXecute** command.
2. Prompt: `Execute what file ?`

   Response: `ACCESS:MEDIAINIT` [Return]
3. Wait for the prompt in step 4 to appear on the screen, then remove the ACCESS: volume and insert the disc you want to initialize.
4. Prompt: `Volume ID ?`

   Response: `#3:` [Return]
5. Prompt: `Are you SURE you want to proceed? (Y/N)`

CAUTION

If the computer reports: `Logical unit #3 - ACCESS:`

in the second line of this display, you are about to initialize the ACCESS: volume. **Remove the ACCESS: disc and insert the new disc to be initialized.**

Response: Press [Y]

6. The following prompt may or may not appear:

   `Formatting option? (defaults to 0)`

   If it does, refer to your disc drive manual for instructions on selecting the correct formatting option. Typical values for double-sided micro-discs are 0 for **LIF**, 3 for **HFS**, and 4 for single-sided micro-discs.

7. Prompt: `Interleave factor? [1..15] (defaults to 2)`

   Response: Press [Return]. (Note: Your prompt may contain different values.)

8. Wait about three minutes for the message: `Volume zeroing completed`. Remove the now initialized disc.

9. If you have another disc to initialize, insert the next disc and press [U]. Begin again at step 4.

10. If the disc is to be an Hierarchical File System (HFS) disc, you need to execute the `MKHFS` program to install the proper directory structure on the disc. (See appendix E of this manual if you have never used HFS discs.)

Additional Information

Flexible disc initialization: 28

Initializing Winchester (Hard) Discs

Volumes on-line: ACCESS:
Subsystem: Main Command Level
Command: **eXecute**

Description

New Winchester (hard) discs must be initialized (formatted) before they can be used. Initialization is performed by a program stored in the file MEDIAINIT.CODE in the ACCESS: volume. The program is run by invoking the **eXecute** command from the Main Command Level.

CAUTION

Only new discs should be initialized. If you want to re-initialize a disc for any reason, be sure to first save its contents on another disc.

Procedure

1. **If you have more than one Winchester disc, turn off all but the ONE you will initialize, and reboot the Pascal system.**
2. Insert the ACCESS: volume into drive 0. Press [X] to invoke the **eXecute** command.
3. Prompt: `Execute what file ?`

   Response: `ACCESS:MEDIAINIT` [Return]
4. Wait for the prompt in step 5 to appear on the screen, then remove the ACCESS: volume.
5. Prompt: `Volume ID ?`

   Response: `#11:` [Return]
6. Prompt: `Are you SURE you want to proceed? (Y/N)`

CAUTION

If the computer reports: `Logical unit #3 - ACCESS:`

in the second line of this display, you are about to initialize the ACCESS: volume. **Remove the ACCESS: volume.** Press [N] to abort the procedure. Press [U] and begin again at step 5.

Response: Press [Y]

7. The following prompt may or may not appear:

   Prompt: `Interleave factor? [1..15] (defaults to 1)`

   Response: [Return]

8. Wait up to *one hour* for the message: `Volume zeroing completed`. The disc is now initialized. Refer to the "Special Instructions" in Chapter 3 if you have an "A" version of the HP 9133, 9134 or 1935 disc drives.

9. If the disc is to be an Hierarchical File System (HFS) disc, you need to execute the `MKHFS` program to install the proper directory structure on the disc. (See appendix E of this manual if you have never used HFS discs.)

Note

If you have a second Winchester disc, refer to the Special Configurations chapter of the *Pascal Workstation System Manual* before using it.

Additional Information

Winchester disc initialization: 28 HFS disc initialization: 306

Specifying Files

Volumes on-line: any
Subsystems: any
Commands: any

Description

File specifications in the Pascal system consist of a **volume ID** or **unit number**, followed by a **file name**. The unit number is the unique number assigned to a particular drive. The volume ID is the name associated with a particular volume. On hierarchical file systems such as HFS or SRM, a file specification may also require a **directory path** before the file name. In other words, the name of a file includes the names of the directories leading up to the file.

File names may have suffixes to designate file type:

- The .CODE suffix indicates Pascal object code, the output of the Compiler, Assembler and Librarian.
- The .TEXT suffix indicates a special form of compacted text file. Includes editing environment.
- The .ASC suffix indicates a LIF ASCII text file.
- The .UX suffix indicates a file that can be shared by the Pascal Workstation and the HP-UX system.
- The .BAD suffix indicates a file covering a damaged area of a disc.
- The .SYSTM suffix indicates a boot image file.
- No suffix indicates a "data" file.

Procedure

To specify a file, type the unit number or volume ID associated with the device or volume containing the file, followed by the file name. To specify an SRM or HFS file, include the file's directory path before the file name. Examples:

- `#3:TEST.TEXT` indicates a file named TEST.TEXT in the volume that resides in unit #3.
- `V11:TEST.ASC` indicates a file named TEST.ASC in the volume with volume ID V11:.
- `ACCESS:MEDIAINIT.CODE` indicates a file named MEDIAINIT.CODE in the volume with volume ID ACCESS:.
- `#5:/USERS/JIM/NOTE.TEXT` indicates a file on the SRM named NOTE.TEXT in the directory JIM. JIM is in the directory USERS, which is in the root directory of the disc. The path specifier for an HFS disc is similar.

Additional Information

Specifying files: 35,42, 296, 319

Finding Flexible Disc Drive Unit Number Assignments

Volumes on-line: ACCESS:
Subsystem: Filer
Command: **Volumes**

Description

Each flexible disc drive in your system is assigned a unique unit number. Use the unit number to select which disc drive you would like to access. Follow this procedure to determine which unit numbers have been assigned to your flexible disc drives.

Procedure

1. Insert the ACCESS: volume into any one of your flexible disc drives.
2. While at the Main Command Level, press [F] to enter the Filer.
3. Press [V] (**Volumes**) to list all on-line volumes on the screen.
4. Find ACCESS: in the right-hand column of the display. The corresponding number in the left-hand column is the unit number for the drive containing the ACCESS: volume.
5. Enter the device name, drive number, and unit number into the System Configuration Table provided at the beginning of this appendix.
6. Insert the ACCESS: volume into another disc drive and begin again at step 3. Repeat this procedure until all drives are identified.

Additional Information

Identifying your flexible disc drives: 46

Finding Winchester Disc Unit Number Assignments

Volumes on-line: ACCESS:
Subsystem: Filer
Command: **Volumes**

Description

A Winchester (hard) disc may contain several volumes. Each volume is assigned a unique unit number. You may use the unit number to select which volume you would like to access. Follow this procedure to determine which unit numbers have been assigned to your Winchester disc. Like all procedures in this book, this one assumes you have only one Winchester disc. If you have others, refer to the Special Configurations chapter of the *Pascal Workstation System Manual.*

Procedure

1. If your Winchester disc is not initialized, refer to the "Initializing Winchester (Hard) Discs" procedure.
2. Insert the ACCESS: volume into any one of your flexible disc drives.
3. While at the Main Command Level, press [F] to enter the Filer.
4. Turn off all Winchester (hard) discs but one.
5. Press [V] (**Vols**) to list all on-line volumes on the screen.
6. All unit numbers listed in the left-hand column that fall within the range 11 through 40 are assigned to your Winchester disc. Each unit number corresponds to one volume on the disc. The right-hand column gives the volume ID (volume name) of each volume.
7. Check that the unit numbers on the Pascal Workstation Worksheet are correct. Update the worksheet if they are not.
8. Press [q] to exit from the Filer.
9. Turn on remaining Winchester discs. Allow them to spin-up, then press [I] (Initialize)
10. Repeat this procedure for each Winchester (hard) disc, starting again at step 3.

Note

If no unit numbers in the range 11 through 40 appear in the list, or if fewer unit numbers have been assigned to the disc than you expected, see the "Special Configurations" chapter of te *Pascal Workstation System* manual.

Additional Information

Identifying your Winchester disc: 48

Finding SRM Unit Number Assignments

Volumes on-line: any
Subsystem: Filer
Command: **Volumes**

Description

The Shared Resource Manager (SRM) appears to the Pascal workstation as two or more volumes. The default volume is usually assigned to unit number five (#5:) while the system volume appears as unit number 45 (#45:). You may use the unit number to select which volume you would like to access. Follow this procedure to determine which unit numbers have been assigned to your SRM. Like all procedures in this book, this one assumes you have access to only one SRM system. If you have others, refer to the Special Configurations chapter of the *Pascal Workstation System Manual.*

Procedure

1. While at the Main Command Level, press [F] to enter the Filer.
2. Press [V] (**Volumes**) to list all on-line volumes on the screen.
3. Find unit number five (#5:) listed in the left-hand column. The volume ID listed in the right-hand column is the default directory.
4. Find unit number forty-five (#45:) listed in the left-hand column. The volume ID listed in the right-hand column is the system directory.
5. Press [Q] to exit the Filer.

Additional Information

Identifying your SRM: 283

Finding HFS Unit Number Assignments

Volumes on-line: any
Subsystem: Filer
Command: **Volumes**

Description

A Hierarchical File System (HFS) disc appears to the Pascal workstation as a single volume (or as two volumes if you booted from it and it is not a removable disc). If it is a Winchester disc, it is usually assigned to unit number eleven (#11:). Follow this procedure to determine which unit number has been assigned to your HFS disc. Like all procedures in this book, this one assumes you have access to only one HFS disc. If you have others, refer to the Special Configurations chapter of the *Pascal Workstation System Manual.*

Procedure

1. While at the Main Command Level, press [F] to enter the Filer.
2. Press [V] (**Volumes**) to list all on-line volumes on the screen.
3. Find unit number eleven (#11:) listed in the left-hand column. The volume ID listed in the right-hand column is the default directory.

   If you booted from the HFS disc, unit number forty-six (#46:) also appears in the volume list.
4. List the volume by using the **Ldir** command. The directory header will indicate the directory structure of the disc.
5. Press [Q] to exit the Filer.

Additional Information

Identifying your HFS disc: 304

Backing Up Your System

Volumes on-line: ACCESS:, all
Subsystem: Filer
Command: **Filecopy**

Description

Back-up copies of each Pascal system disc should be made immediately to protect your software investment. You will need one initialized disc for each Pascal system disc.

Procedure

1. Write-protect all of your Pascal system discs (see Chapter 3).
2. Insert the ACCESS: volume into unit #3. While at the Main Command Level, press [F] to enter the Filer. Wait for the Filer's command line to appear at the top of the screen.
3. If you have two flexible disc drives, insert a newly initialized disc into the other drive.
4. Press [F] to invoke the **Filecopy** command.
5. Prompt: `Filecopy what file ?`

   Response: `#3:` [Return]
6. Prompt: `Filecopy to what ?`

   Response:

   - **If you have two flexible disc drives,** type the unit number of the second disc drive that contains the newly initialized disc, and press [Return]. For example, if your second drive is assigned unit #4, you would type: `#4:` [Return]
   - **If you have one flexible disc drive,** type: `#3:` [Return]
7. If you have only one disc drive, wait for the following prompt, then remove the ACCESS: disc, insert a newly initialized disc into unit #3, and press [C]:

```
Please mount DESTINATION in unit #3
'C' continues, <esc> aborts
```

8. Prompt: `Destroy EVERYTHING on volume V3 ? (Y/N)`

   Response:

   - If the directory name shown above is the name of the Pascal system volume you are copying, press [N] to abort the operation. Start over, making sure you exchange discs in step 7 (single disc drive systems), or enter the unit numbers of the source and destination volumes correctly (multiple disc drive systems).
   - If the directory name is V3, V4, V7 or something similar, press [Y] to proceed.

9. When the Filer's command line reappears, remove the new copy and label it. Remove ACCESS: and insert the next Pascal system disc to be copied into unit #3. Begin again at step 3, and repeat this procedure until you have made copies of all Pascal system discs.

Additional Information

Backing up your system: 51, 333

Copying Pascal System Files to a Winchester (Hard) Disc

Volumes on-line: BOOT:, or BOOT2:, ACCESS:
Subsystem: Filer
Command: **Filecopy**

Description

This procedure describes how to copy Pascal boot files, subsystems and utility programs to a Winchester (hard) disc. Before establishing your final Winchester disc configuration, however, you must read the Special Configurations chapter of the *Pascal Workstation System Manual.* This procedure assumes there was only one Winchester disc on-line when the Pascal system was booted.

If you will be using your hard disc as an **Hierarchical File System** (HFS) see Appendix E of this manual for an alternate procedure.

Procedure

Note

If your computer has Boot ROM 3.0L, 2.0 or 1.0, you cannot boot Pascal from a Winchester disc.

1. Insert the ACCESS: volume into one of your disc drives. While at the Main Command Level, press [F] to load the Filer.
2. When the Filer's command line appears at the top of the screen, remove the ACCESS: volume and insert the BOOT: (or BOOT2:) volume. Press [F] to invoke the **Filecopy** command.
3. Prompt: `Filecopy what file ?`

   Response: Type either `BOOT:=` or `BOOT2:=` and press [Return]
   Your response depends upon which disc (BOOT: or BOOT2:) you are copying.
4. Prompt: `Filecopy to what ?`

   Response: `#11:$` [Return]

5. Each file in the BOOT: volume is copied, one at a time, to the first volume of the Winchester disc.
6. Refer to the "Copying Multiple Files" procedure later in this appendix to *selectively* copy Pascal subsystems, utility programs, and libraries to your Winchester disc. At the very least, you will probably want to copy the Filer, Editor and Compiler. **Never use the "Copying an Entire Volume" procedure to copy a flexible disc volume to a Winchester disc.**

Additional Information

Copying the boot files: 172
Copying the Pascal subsystems: 175
= wildcard: 173, 253, 256
$ wildcard: 152, 174, 176, 206, 252, 253

Permanently Loading Subsystems

Volumes on-line: any
Subsystem: Main Command Level
Command: **Permanent**

Description

Rapid movement between subsystems can be attained by permanently loading the most frequently-used subsystems into memory, using the Main Command Level's **Permanent** command. Refer to Appendix C for subsystem memory requirements and decide which subsystems to permanently load.

Procedure

1. Put the volume on-line that contains the subsystem you want to permanently load.
2. At the Main Command Level, press [P] to invoke the **Permanent** command.
3. Prompt: `Load what code file?`

   Response:
   Type the file specification for the subsystem you want to permanently load. If the file does not have the .CODE suffix (e.g., FILER, EDITOR, COMPILER), follow the file name with a period. For example, to permanently load the Filer, your response would be:

   `ACCESS:FILER.` [Return]
4. When the command line reappears at the top of the screen, the subsystem is permanently loaded.

Additional Information

Permanent command: 120, 178
Permanently loading subsystems: 178
Listing a directory: 148
Specifying a file: 35, 42
System memory requirements: 275

Making a Memory Volume

Volumes on-line: None
Subsystem: Main Command Level
Command: **Memvol**

Description

A memory volume with volume ID RAM: can be created to serve as a high-speed mass storage device. Since the RAM: volume displaces available program space, memory size is a consideration. The following procedure creates a LIF RAM: volume at unit #50, with a size of 200 512-byte blocks and 8 directory entries. You may choose parameters to suit your own situation.

Procedure

1. At the Main Command Level, press [M] to invoke the **Memvol** command.
2. Prompt: `What unit number?`

   Response: `#50:` [Return]
3. Prompt: `How many 512 byte BLOCKS?`

   Response: `200` [Return]
4. Prompt: `How many entries in directory?`

   Response: `8` [Return]
5. The computer creates the RAM: volume and zeroes it.

See `MKHFS` in Appendix E to convert the memory volume to HFS format.

Additional Information

Memvol command: 123
Making a RAM: volume: 180

Setting the System Volume

Volumes on-line: None
Subsystem: Main Command Level
Command: **What**

Description

The system volume is used for storing temporary system files during compilation, storing workfiles, storing the system date, etc. The system volume must usually have considerable free space, and should have a fast access time. The system volume need not be changed from its initial setting. However, if you will be using workfiles (discussed later) and have a memory volume, it may be convenient to change the system volume to RAM:.

Procedure

1. At the Main Command Level, press [W] to invoke the **What** command.
2. A table showing the file specifications of all subsystems appears on the screen. Press [S] to signal that you want to change the system volume entry (toward the bottom of the screen).
3. Type the volume ID, the unit number, or directory path (for HFS and SRM systems) of the new system volume and press [Return]. For example, to set the system volume to RAM:, type:

   `RAM:` [Return]

4. Press [Q] to exit from the **What** command.

Note

The system volume may be accessed using its volume ID (e.g., RAM:), its unit number (e.g., #50:), or by the special symbol *. For example, the file specification `*WORK.TEXT` indicates a file named WORK.TEXT in the system volume.

Note

When setting up an HFS flexible disc as the system volume, **do not** provide path information. HFS flexible discs are not allowed to be prefixed below their root (/) directories (see the appendix "The Hierarchical File System").

Additional Information

Setting the system volume: 118, 125, 181, 196
System volume considerations: 182
What command: 117, 181
Newsysvol command: 125, 196

Setting the Default Volume

Volumes on-line: None
Subsystem: Main Command Level
Command: **What**

Description

The default volume is the volume ID assumed if none is explicitly given when specifying a file. For example, if the default volume is ACCESS:, then the file specification MEDIAINIT.CODE is equivalent to ACCESS:MEDIAINIT.CODE; the default volume ACCESS: is assumed in the former. The default volume should be set to the most frequently-accessed volume, to spare you from typing the volume ID every time you access a file.

Procedure

1. At the Main Command Level, press [W] to invoke the **What** command.
2. A table showing the file specifications of all subsystems appears on the screen. Press [D] to signal that you want to change the default volume entry at the bottom of the screen.
3. Type the volume ID, the unit number, or directory path (for HFS and SRM systems) of the new default volume and press [Return]. For example, to set the default volume to ACCESS:, type:

   ACCESS: [Return]

4. Press [Q] to exit from the **What** command.

Note

The default volume may be accessed in these three ways, by using its volume ID (e.g., ACCESS:), its unit number (e.g., #3:), or by the special symbol :.

Note

When prefixing to an HFS flexible disc, **do not** provide path information. HFS flexible discs are not allowed to be prefixed below their root (/) directories (see the appendix "The Hierarchical File System").

Note

You may set the default volume to a disc drive (unit) instead of to a specific volume by entering the unit number of an empty drive. *Any* volume you insert into the drive is then designated the default volume.

Additional Information

Setting the default Volume: 118, 155, 183
What command: 117, 181
Prefix-vol command: 155, 183

Creating an Autostart File

Volumes on-line: ACCESS:
Subsystem: Editor
Command: Various Editing Commands

Description

An Autostart file will automatically set up your programming environment every time you boot the Pascal system. The Autostart file must be stored on the system volume assigned after the Pascal system is booted.

Procedure

The procedure for creating an Autostart file is lengthy and highly individual. Refer to Chapter 10 for instructions (see page reference below).

Additional Information

Creating a new Autostart file: 184, 209

Creating/Editing a File

Volumes on-line: ACCESS:
Subsystem: Editor
Command: Various Editing Commands

Description

Files are created and modified in the Editor subsystem. The Editor is placed into various modes of operation with the following commands:

- **Insert mode:** press [I] to type in text.
- **Delete mode:** press [D] to delete text.
- **Exchange mode:** press [X] to type new characters over old ones.
- **Copy File mode:** press [C] [F] to copy a file into the Editor.
- **Copy Buffer mode:** press [C] [B] to copy the buffer. The buffer contains the text of your last deletion or insertion.
- **Set Document:** press [S] [D] to set the editing parameters for document editing.
- **Set Program:** press [S] [P] to set the editing parameters for program editing.
- **Margin:** press [M] to clean up a paragraph (only works in the document editing environment).

The command line for each mode gives exiting options. Most modes are exited with [Select] to keep the changes made while in the mode, or [ESC] to abort the changes made while in the mode.

Procedure

1. With the ACCESS: volume on-line, press [E] to invoke the Editor.
2. Press [Return] to create a new file, or type the file specification of the file you want to edit.
3. Use the Editor's commands to insert and edit text.
4. Press [Q] to exit the Editor. Refer to the next procedure, "Storing a File (Editor)," for exiting options.

Additional Information

Editing commands: 129
Keyboard cursor control: 82, 96, 107

Storing a File (Editor)

Volumes on-line: ACCESS:, destination volume
Subsystem: Editor
Command: **Quit**

Description

The Editor is exited by pressing [Q] to invoke the **Quit** command. Several exiting options are then presented:

- **Update the workfile and leave.** Press [U] to write the edited file to the workfile WORK.TEXT in the system volume.
- **Exit without updating.** Press [E] to exit the Editor and discard the edited file.
- **Return to the Editor without updating.** Press [R] to return to the Editor without writing the file.
- **Write to a file name and return.** Press [W] to write the edited file to the file specification of your choice.
- **Save as file new file....** Press [S] to write the edited file using the same file specification that was used to retrieve it. The old version of the file is removed. This option appears only when you enter the Editor with an existing file.
- **Overwrite as file....** Press [O] to write the edited file directly over the old version of the file, keeping the same file name. This option appears only when you enter the Editor with an existing file. Use this only on HSF and SRM systems, or if the newly edited version of the file is not larger than the old version.

  On HFS and SRM systems, this option updates the file such that all duplicate links to the file point to the updated copy of the file. (In contrast, choosing the **Save** option updates the file such that the path you specify points to the updated version of the file, but any other duplicate links to the file remain set to the original (old) copy of the file.)

Procedure

1. Press [Q] to quit the Editor.
2. Select one of the exiting options above to store the edited file. Make sure the volume where you want to store the file is on-line.
3. Some options allow you to return to the Editor after storing the edited file. Press [R] to return to the Editor, or [E] to exit from the Editor.

Additional Information

Editor commands: 129
Exiting the Editor: 59, 140
Workfiles: 144, 146, 166, 193

Creating Workfiles (Editor)

Volumes on-line: ACCESS:, system volume
Subsystem: Editor
Command: **Quit** with Update

Description

An edited file can be stored into a file named WORK.TEXT in the system volume. WORK.TEXT is called a workfile. The workfile behaves like the "default" file. If a workfile exists, it will automatically be read into the Editor for editing, and will automatically be compiled when you enter the Compiler. This special treatment speeds program development by sparing you from typing file specifications as you move between subsystems. This procedure describes how to create a workfile in the Editor.

Procedure

1. Enter the Editor, edit your file, and press [Q] to exit the Editor.
2. Make sure the system volume is on-line and press [U] to update the workfile. The edited file is written into the file WORK.TEXT in the system volume.
3. If you wish, press [E] to re-enter the Editor. Notice how the Editor automatically reads in the workfile.

Additional Information

Workfiles: 59, 140
Exiting the Editor: 144, 146, 166, 193

Creating Workfiles (Filer)

Volumes on-line: ACCESS:, source volume, system volume
Subsystem: Filer
Command: **Get**

Description

An existing file can be made the workfile with the Filer's **Get** command. **Get** associates the file name you specify with the workfile.

Procedure

1. Enter the Filer and press [G] to invoke the **Get** command.
2. The following prompt may appear if a workfile already exists:

   ```
   Throw away current workfile ? (Y/N)
   ```

   If the workfile contains valuable information that you want to keep, press [N] and use the **Save** command to save it (see the "Storing Workfiles (Filer)" procedure). Otherwise, press [Y] to discard it.
3. Prompt: `Get what file?`

   Response:
   Make sure the volume containing your file is on-line, and type the file specification for the file you wish to designate the workfile. Press [Return]. For example, if you wanted to designate a file named TEST.TEXT in the volume in unit #3 as the workfile, you would type:

   `#3:TEST` [Return]

   The suffix, .TEXT, is automatically appended by the **Get** command. To suppress the suffix, type a period after the file name.

Additional Information

Get command: 146, 166
Workfiles: 144, 146, 166, 193

Storing Workfiles (Filer)

Volumes on-line: ACCESS:, system volume, destination volume
Subsystem: Filer
Command: **Save**

Description

Workfiles can be copied from the system volume to another volume with the Filer's **Save** command. If possible, the **Save** command will save both WORK.TEXT and WORK.CODE at the same time.

Procedure

1. Enter the Filer and press [S] to invoke the **Save** command.
2. If the workfile was created with the **Get** command, the computer will ask if you want to save the workfile into the same file you retrieved it from:

   ```
   Save as ... (Y/N)
   ```

   Press [Y] if you do, otherwise press [N].
3. If the workfile was created when exiting the Editor, or if you do not want to save the workfile with its original file name, the computer will ask for a file specification:

   ```
   Save as what file?
   ```

   Type the file specification for the new file and press [Return]. For example, if you want to store the workfile into a file named TEST.TEXT in a volume named PROG:, you would type:

   ```
   PROG:TEST [Return]
   ```

   The suffix, .TEXT, is automatically appended by the **Save** command. To suppress the suffix, type a period after the file name.

Additional Information

Save command: 147, 193, 204
Workfiles: 144, 146, 166, 193

Clearing a Workfile

Volumes on-line: ACCESS:, system volume
Subsystem: Filer
Command: **New**

Description

In many cases, workfiles must be cleared before the subsystems will operate on another file. For example, as long as a workfile exists, you cannot edit or compile another file. The Filer's **New** command removes the workfile(s) from the system volume.

Procedure

1. Enter the Filer and press [N] to invoke the **New** command.
2. If the workfile has not been stored in another volume with the **Save** command, the computer will ask:

   ```
   Throw away current workfile ? (Y/N)
   ```

   Respond with [Y] to discard it, or press [N] and use **Save** to store it.
3. The workfile is now cleared.

Additional Information

New command: 157
Save command: 147, 193, 204
Workfiles: 144, 146, 166, 193

Copying a Single File

Volumes on-line: ACCESS:, source and destination volumes
Subsystem: Filer
Command: **Filecopy**

Description

The following procedure copies a single file from a source volume to a destination volume.

Procedure

1. Enter the Filer and press F to invoke the **Filecopy** command.
2. Prompt: `Filecopy what file ?`

   Response:
   Type the file specification of the file you want to copy and press Return. Make sure the volume containing the file is on-line.
3. Prompt: `Filecopy to what ?`

   Response:

   - To give the new copy of the file a new name, type the volume ID (or unit number) of the destination volume, followed by the new file name (Example: `#3:NEWNAME.ASC`). Press Return.
   - To give the new copy of the file the same name as the original, type the volume ID (or unit number) of the destination volume, followed by `$` (Example: `#3:$`). Press Return. The `$` is a wildcard that means "same name."
4. The file is copied. If the source and destination units are the same, you will be prompted at the appropriate time to insert the destination volume. Exchange discs and press C.

Additional Information

Filecopy command: 52, 54, 151, 173, 175, 205
Copying a single file: 152, 176
$ wildcard: 152, 174, 176, 206, 252, 253

Copying Multiple Files

Volumes on-line: ACCESS:, source and destination volumes
Subsystem: Filer
Command: **Filecopy**

Description

The following procedure copies several files from a source volume to a destination volume, allowing you to select which files will be copied and which will not. Each new copy will receive the same name as the original. To change the names of the destination files, refer to the discussion of wildcards in the *Pascal Workstation System Manual.*

Procedure

1. Enter the Filer and press [F] to invoke the **Filecopy** command.
2. Prompt: `Filecopy what file ?`

   Response:
   Make sure the source volume is on-line, and type the volume ID (or unit number) of the source volume where the original files reside, followed by `?` (Example: `#3:?`). Press [Return]. The `?` is a wildcard that will cause the computer to prompt you before each file is copied. You can then confirm or reject each file.
3. Prompt: `Filecopy to what ?`

   Response:
   Type the volume ID (or unit number) of the destination volume, followed by `$` (Example: `V4:$`). Press [Return]. The `$` is a wildcard that means "same name."
4. File copying now commences. To copy a file, press [Y] when prompted. To reject a file, press [N]. If the source and destination units are the same, you will be prompted at the appropriate time to insert the destination volume; exchange discs and press [C].

Additional Information

Filecopy command: 52, 54, 151, 173, 175, 205
Copying multiple files: 175
$ wildcard: 152, 174, 176, 206, 253
? wildcard: 175, 257

Copying All Files in a Volume or Directory

Volumes on-line: ACCESS:, source and destination volumes
Subsystem: Filer
Command: **Filecopy**

Description

The following procedure copies all files, one at a time, from a source volume to a destination volume. Each new copy will receive the same name as the original. To change the names of the destination files, refer to the discussion of wildcards in the *Pascal Workstation System Manual.*

Procedure

1. Enter the Filer and press [F] to invoke the **Filecopy** command.
2. Prompt: `Filecopy what file ?`

   Response:
   Type the volume ID (or unit number) of the source volume where the original copies reside, followed by = (Example: `#3:=`). Press [Return]. Make sure the source volume is on-line. The = is a wildcard that will cause the computer to read each file in the source volume, one at a time.
3. Prompt: `Filecopy to what ?`

   Response:
   Type the volume ID (or unit number) of the destination volume, followed by **$** (Example: `V4:$`). Press [Return]. The **$** is a wildcard that means "same name."
4. File copying now commences. Files are copied from the source volume to the destination volume, one at a time. If the source and destination units are the same, you will be prompted at the appropriate time to insert the destination volume. Exchange discs and press [C].

Additional Information

Filecopy command: 52, 54, 151, 173, 175, 205
Copying all files in a volume: 172
$ wildcard: 152, 174, 176, 206, 252
= wildcard: 173, 257

Copying An Entire Volume

Volumes on-line: ACCESS:, source and destination volumes
Subsystem: Filer
Command: **Filecopy**

Description

The following procedure copies the entire source volume, all at once, to the destination volume. Both source and destination volumes will be identical.

Note

Do not use this procedure to copy a small source volume to a large destination volume. The destination volume will become the same size as the source, making much of its storage capacity inaccessible.

You can **not** copy an entire SRM volume with this method. (See appendix D for SRM copying instructions.) See Appendix E for auxiliary HFS copying instructions

---

Procedure

1. Enter the Filer and press [F] to invoke the **Filecopy** command.
2. Prompt: `Filecopy what file ?`

   Response:
   Type the volume ID (or unit number) of the source volume (Example: `#3:`). Press [Return]. Make sure the source volume is on-line.
3. Prompt: `Filecopy to what ?`

   Response:
   Type the volume ID (or unit number) of the destination volume (Example: `V4:`). Press [Return].
4. Volume copying now commences. If the source and destination units are the same, you will be prompted at the appropriate time to insert the destination volume. Exchange discs and press [C].

Additional Information

Removing a Single File

Volumes on-line: ACCESS:, source volume
Subsystem: Filer
Command: **Remove**

Description

A file can be removed from a volume with the Filer's **Remove** command.

Procedure

1. Enter the Filer and press [R] to invoke the **Remove** command.
2. Prompt: `Remove what file?`

   Response:
   Make sure the volume containing the file to be removed is on-line, then type the *complete* file specification and press [Return] (Example: `#3:TEST.TEXT`).
3. The file is removed.

Additional Information

Remove command: 153
File specifications: 35, 42

Removing Multiple Files

Volumes on-line: ACCESS:, source volume
Subsystem: Filer
Command: **Remove**

Description

Multiple files can be removed from a volume with the Filer's **Remove** command, and the wildcards `?` and `=`.

Procedure

1. Enter the Filer and press [R] to invoke the **Remove** command. Make sure the volume containing the file to be removed is on-line.
2. Prompt: `Remove what file?`

   Response:

   - To remove *all* files from a volume, type the volume ID (or unit number) of the volume, followed by `=` (Example: `V3:=`). Press [Return].
   - To *choose* which files in the volume are to be removed, type the volume ID (or unit number) of the volume, followed by `?` (Example: `#3:?`). Press [Return].

3. The files in the volume will be listed on the screen one at a time (if you used `?`) or all at once (if you used `=`). If removing all files, press [Y] to proceed with the remove, or [N] to abort. If choosing which files to remove, press [Y] to remove a file, or [N] to keep a file. After choosing which files will be removed, review the list of files on the screen and press [Y] to proceed with the remove, or [N] to abort the operation.

Additional Information

= wildcard: 174
? wildcard: 176
Remove command: 153

Changing a File Name

Volumes on-line: ACCESS:, source volume
Subsystem: Filer
Command: **Change**

Description

The following procedure changes the name of a file in the source volume.

Procedure

1. Enter the Filer and press [C] to invoke the **Change** command.
2. Prompt: `Change what file ?`

   Response:
   Type the file specification of the file whose name you want to change and press [Return]. For example, to change the name of the file FRED.TEXT in the volume in unit #3, you would type:

   `#3:FRED.TEXT` [Return]

4. Prompt: `Change to what ?`

   Response:
   Type the new name of the file and press [Return]. Do not type the volume ID (or unit number) again. For example, to change the file to GEORGE.TEXT, you would type:

   `GEORGE.TEXT` [Return]

5. The file's name has now been changed.

Additional Information

Change command: 152

Changing a Volume Name

Volumes on-line: ACCESS:, source volume
Subsystem: Filer
Command: **Change**

Description

The following procedure changes the name (volume ID) of a volume.

Procedure

1. Enter the Filer and press C to invoke the **Change** command.
2. Prompt: `Change what file ?`

   Response:
   Type the volume ID or unit number of the volume whose name you want to change and press Return. For example, to change the name of the volume V3:, you would type:

   `V3:` Return

4. Prompt: `Change to what ?`

   Response:
   Type the new name (volume ID) of the volume (up to six letters). Press Return. For example, to change the volume to MEMOS:, you would type:

   `MEMOS:` Return

5. The volume's name has now been changed.

Additional Information

Change command: 152

Changing a Directory Name

Volumes on-line: ACCESS:, source volume
Subsystem: Filer
Command: **Change**

Description

The following procedure changes the name of an SRM or HFS directory.

Procedure

1. Enter the Filer and press C to invoke the **Change** command.
2. Prompt: `Change what file ?`

   Response:
   Type the volume number and name of the directory whose name you want to change and press Return. For example, to change the name of `/USERS/JOE` to `/USERS/FRED`, you would type:

   `/USERS/JOE` Return

4. Prompt: `Change to what ?`

   Response:
   Type the new name of the directory. Press Return. For example, to change the directory to `FRED`, you would type:

   `FRED` Return

5. The directory's name has now been changed.

Additional Information

Change command: 152

Listing a Directory

Volumes on-line: ACCESS:, source volume
Subsystem: Filer
Command: **Ldir**

Description

The following procedure lists the directory of the source volume, providing a variety of information about each file in the volume.

Procedure

1. Enter the Filer and press [L] to invoke the **Ldir** command. Make sure the volume whose directory you want to list is on-line.
2. Prompt: `List what directory ?`

   Response:
   Type the volume ID (or unit number) of the volume whose directory you want to list. Press [Return]. For example, to list the directory of the volume in unit #3, you would type:

   `#3:` [Return]
3. All files in the directory are listed on the screen. If the directory is too long to fit on the screen, the computer will pause. Press the space bar to view the rest of the listing.
4. To list the files in alphabetical order, type `[*]` following the volume specification (Example: `#3:[*]`).

Additional Information

Ldir command: 148, 165
Interpreting a directory listing: 149

Listing All Volumes On-line

Volumes on-line: ACCESS:, any
Subsystem: Filer
Command: **Volumes**

Description

The Filer's **Volumes** command lists all on-line volumes on the screen, along with their associated unit numbers.

Procedure

1. Enter the Filer and press [V] to invoke the **Volumes** command.
2. All on-line volumes are listed.

- The left-hand column gives the unit number for each volume
- The right-hand column gives the volume ID of each volume.
- A # between the unit number and volume ID indicates a blocked volume (one capable of storing files).
- A * between the unit number and volume ID indicates the system volume.
- Unblocked units are also listed: unit #1 (CONSOLE:) is assigned to the screen; unit #2 (SYSTERM:) is assigned to the keyboard; unit #6 (PRINTER:) is assigned to the printer.
- The current prefix (default) volume is given at the bottom of the display.

Additional Information

Volumes command: 46, 156
Volumes: 35
Unit numbers: 36, 42
System volume: 118, 125, 181, 196
Default (Prefix) volume: 118, 155, 183

Compacting Volume Space

Volumes on-line: ACCESS:, source volume
Subsystem: Filer
Command: **Krunch**

Description

As files are stored to, and deleted from, a LIF volume, free space may become fragmented. When this happens, there may not be a single contiguous group of free blocks large enough to contain a file. An error message, `No room on vol`, will be reported when you try to store a file in such a volume. The solution is to use the Filer's **Krunch** command to compact all free space into one contiguous "hole" at the end of the volume. It is good practice to back up a disc before invoking **Krunch**, for if the power were to fail during the process, files on the disc may be lost.

You can **not** use this command on HFS or SRM discs.

Procedure

1. Enter the Filer and press [K] to invoke the **Krunch** command. Make sure the volume you want to crunch is on-line.
2. Prompt: `Crunch what directory ?`

   Response:
   Type the volume ID (or unit number) of the volume you want to crunch. Press [Return]. For example, if you want to crunch the volume in unit #3, type:

   `#3:` [Return]
3. Prompt: `Crunch directory ... ? (Y/N)`

   Response:
   Press [Y] if the correct volume is given in the prompt; press [N] if not.
4. The volume is then crunched. Do not try to abort the crunch operation. If you do, some, if not all, of the files on the volume may be lost.
5. The crunch operation is finished when the following prompt is displayed:

   `Crunch completed.`

Additional Information

Krunch command: 154
"No room on vol" message: 154

Compiling a Pascal Program

Volumes on-line: CMP:, source volume
Subsystem: Compiler
Command: **Compiler**

Description

The following procedure describes how to compile a Pascal program.

Procedure

1. Make sure the CMP: (or CMP20: for Model 320 computers) volume is on-line and press [C] from the Main Command Level to load and run the Compiler.
2. Prompt: `Compile what text?`

   Response:
   Type the file specification of the file containing the program you want to compile. Press [Return]. For example, if the program resides in a file named PROG.TEXT in the *default* volume, you would type:

   `PROG` [Return]

   The Compiler automatically appends the suffix, .TEXT. To suppress the suffix, type a period after the file name. This prompt will not appear if a workfile exists; the computer assumes you want to compile the workfile and skips to the next prompt.
3. Prompt: `Printer listing (l/y/n/e) ?`

   Response:

- Press [L] to send the program listing to a file. You must then give the file specification of the listing file.
- Press [Y] to send the program listing to the printer.
- Press [N] to produce no program listing. Any errors will be displayed on the screen.
- Press [E] to send all reported errors to the printer *only.*

4. Prompt: `What listing file?`

   Response:
   This prompt appears only if you selected the "L" option in the previous prompt. Type the file specification of the file where you want the program listing put, followed by `[x]`, where x is the size of the listing file (in blocks). For example, to create a listing file named LIST.TEXT in the volume V3: that is 25 blocks long, type:

   `V3:LIST[25]` Return

   The suffix, .TEXT, is automatically appended. To suppress the suffix, type a period after the file name.

5. Prompt: `Output file (default is "...CODE") ?`

   Response:
   Press Return to send the compiler output to the default code file. Otherwise, type the file specification of the file where you want the output placed. This prompt does not appear if a workfile exists; the code is automatically put into the file WORK.CODE in the system volume.

6. Compilation is now performed.

Additional Information

Using the Compiler: 161, 194, 200, 203

Running Programs

Volumes on-line: source volume
Subsystem: Main Command Level
Commands: **Run**, **eXecute**, **User**

Description

The Main Command Level's **Run**, **eXecute** and **User** commands can all be used to run a program under different circumstances.

Procedures

eXecute

To load and run any compiled program that is stored in a file, press [X] to invoke the **eXecute** command. The following prompt will appear:

```
Execute what file ?
```

Type the file specification of the file containing the program, and press [Return]. For example, to run a program stored in a file named PROG.CODE in unit #3, you would type:

```
#3:PROG [Return]
```

The suffix, .CODE, is automatically appended. If the file does not have a .CODE suffix, type a period after the file name.

Run

If an uncompiled workfile exists, press [R] to invoke the **Run** command. The file WORK.TEXT will be compiled, loaded and run. If the workfile WORK.CODE exists, the compilation step is skipped and the workfile is immediately run. If no workfile exists, **Run** behaves just like **eXecute**.

User

To re-run the last program that was run, press [U] to invoke the **User** restart command. Pascal subsystems are programs also, and can be re-entered by pressing [U].

Additional Information

eXecute command: 28, 115
Run command: 119, 201, 203
User command: 32, 120

Printing a File

Volumes on-line: ACCESS:, source volume
Subsystem: Filer
Command: **Translate**

Description

This procedure describes how to print a file on the printer.

Procedure

1. Enter the Filer and press [T] to invoke the **Translate** command.
2. Make sure the volume containing the file you want to print is on-line. Make sure the printer is connected, turned on, and on-line.
3. Prompt: `Translate what file ?`

   Response:
   Type the file specification of the file you want to print and press [Return]. For example, if you want to print the file LETTER.TEXT in the volume NOTES:, you would type: `NOTES:LETTER.TEXT` [Return]
4. Prompt: `Translate to what ?`

   Response for local printer: `#6:` [Return]
   Where #6: is the unit number of the printer connected to your computer. If you prefer, you may use the volume ID PRINTER: in place of unit #6.

   Response for SRM printer:
   If your workstation is on an SRM system, translate the file into an ASCII-type file in a shared printer's spooler directory. (Ask your SRM system manager for the name of each shared printer's spooler directory.) For example, type:

   `#5:/lp/FFLETTER.ASC` [Return]

   The computer translates LETTER.TEXT into an ASCII format for printing on a shared printer and places the translated copy of the file in the printer's spooler directory ("lp" in this example). The "FF" at the beginning of the file name prevents the printer from writing over paper perforations.
5. The file is printed on the printer. If this doesn't work, refer to the Special Configurations chapter of the *Pascal Workstation System Manual.*

Additional Information

Translate command: 157

Volumes on-line:
Subsystem:
Command:

Description

Procedure

Additional Information

Volumes on-line:
Subsystem:
Command:

Description

Procedure

Additional Information

Glossary

B

block A block is the smallest unit of mass storage accessed by a file system. LIF discs use 256-byte blocks. WS1.0 volumes use 512-byte blocks. HFS volumes usually use 1024-byte blocks and SRM volumes use 1-byte blocks.

boot device The peripheral where the Boot ROM found and loaded the Pascal operating system. The Boot ROM has a search pattern which allows booting from just about any drive in any HP mass storage product, including the Shared Resource Manager.

control character Any ASCII character whose value is either 127 or in the range of 0 thru 31. Use of control characters in the Editor and Filer is discouraged, because they may have undesirable effects.

cursor The underline (_) symbol on the screen. The cursor functions as a reference point for the Editor commands which manipulate text and as a reference point for prompts in other Pascal subsystems.

directory Contains information about the files on a volume. This information includes the volume name and the following information about each file on the medium: the file name, the file size (in number of bytes or blocks), the date of last modification to the file, and the file type (which reflects the file's attributes). On Hierarchical File System (HFS) discs and on the Shared Resource Manager (SRM), a file type of "Directory" indicates that another directory exists "below" the current directory. Directory information can be seen by using the Filer's List Directory and Extended Directory commands. The directory is initialized with the Filer's Zero command.

dollar sign This character "$" is used in the Filer as a convenience in specifying file names. When used in place of a destination file name, it means that the file is to have the same name as the source file.

environment The conditions or parameters which affect how text in the Editor is Adjusted, Inserted, and Margined. These parameters may be changed with the Editor's **Set** command.

file A discrete collection of information designated by a file name and residing on a mass storage medium.

file name An entry in a directory which identifies a particular file.

file specification Completely identifies a file and may include both a volume specification and a file name. A volume specification can be one of many items, but it is always part of a file specification. If a volume ID is given, it must be separated from the file name by a colon (:). If not, the default volume is assumed.

file types Several file types are recognized by the Pascal Workstation. File generally (but not always) have a suffix as part of the file name which indicates their type. The file type is established at the time of the file's creation and cannot be changed just by changing the suffix. The types and their associated suffixes are:

- TEXT files - (suffix is .TEXT) Contain ASCII characters and Editor environment information.
- ASCII files - (suffix is .ASC) Are similar to TEXT files. The format is slightly different and there is no Editor environment information.
- HP-UX files - (suffix is .UX) Are files designed to be easy to exchange among the BASIC, Pascal, and HP-UX workstations. They are compatible with HP-UX "regular" files.
- CODE files - (suffix is .CODE) Contain code generated by the Pascal Assembler, Compiler or Librarian.
- Data files - (no specific suffix) Are files which can be created by an subsystem but are used primarily as INPUT and OUTPUT files in Pascal programs. They do not have suffixes.
- System files - (suffix is .SYSTM) Are files created with the Librarian's Boot command. They are loadable by the Boot ROM except on HFS volumes.
- Bad files - (suffix is .BAD) Are a type of file created by the user to isolate unreliable or worn-out areas on a mass storage medium. Once created, BAD files will not be moved by subsequent crunches of the volume.

Librarian A Pascal subsystem designed to manage object modules. It can link or just collect object modules together into object files. The Librarian is the file named LIBRARIAN in the operating system and is accessed by pressing **L** from the Main Command Level.

Main Command Level The level from which all the subsystems of the Pascal Workstation are entered. The prompt displayed at this level looks like:

```
Command: Compiler Editor Filer Initialize Librarian Run eXecute Version ?
```

mouse A small, rodent-like input device, consisting of a roller ball and buttons. Rolling the device on any surface generates two-dimensional movement information that is transmitted through its tail to the computer. Pushing the buttons also generates information that is sent to the computer. The mice available with Series 200/300 equipment are connected to the computer through the HP Human-Interface Link (HP-HIL).

on-line Any object (device, volume or file) current accessible by the Pascal Workstation.

pathname On hierarchical file systems, the names of the directories and sub-directories leading up to and including the name of a file. For example, `/users/mike/projects/plan.text` is a pathname where the path is `/users/mike/projects` and the file name is `plan.text`.

peripheral An I/O device such as a printer or disc.

prompt Generally, any request for information from the system. The different Pascal subsystems have primary prompts (the Editor Prompt, Filer Prompt, etc.) and many subsystem commands have prompts of their own which are displayed at the top of the screen when the command is entered.

string A contiguous series of non-control ASCII characters.

system volume or system unit The Pascal Workstation distinguishes one mass storage unit to be used for special purposes. This "system volume" is where the date and any AUTOSTART file are found at boot time. It is where the system looks first for system files such as the Compiler and Editor, where workfiles are stored, and where an intermediate file is stored during interpretation of a Stream (command) file.

text file A file created and/or used by the Editor which contains ASCII or selected foreign language characters. The Editor automatically appends `.TEXT` to a file name unless it either already contains a suffix or the last character in the name is a period. A text file may be of type TEXT, ASCII, UX or DATA.

unit An entry in the Unit Table.

unit number An integer in the range from 1 through 50 representing the volume having the corresponding entry in the unit table.

volume A volume refers to any I/O device such as a printer, keyboard, screen or mass storage device. The name of a mass storage volume is found in its directory; the name of an unblocked device is found in its Unit Table entry. There may be several volumes on one physical storage medium. Hard discs typically contain multiple volumes (if they use LIF), but flexible discs generally have only a single volume. The volume may be mounted (in a disc drive) or not. The syntax of a volume name depends on its type (for example, LIF and HFS volume names may contain 6 characters, WS1.0 may contain 7, and SRM may contain 16).

wildcard Both the characters = and ? can be used in the Filer as wildcards in place of parts of a file specification.

workfile If the workfile exists, it is the automatic file used by the Editor, Compiler, Assembler, Debugger and the Run command. It is designated when quitting the Editor using the Update option or the Filer's Get command.

Pascal System Components C

Subsystem Memory Requirements

The table below shows the approximate amount of memory consumed by each subsystem when it is permanently loaded (P-loaded). Additional memory may be required when a subsystem is run, proportional to the size of the file it is operating on.

Subsystem Name	Memory Consumed by P-loading (K bytes)[1]
Assembler	111
Compiler	221
Editor	54
Filer	55
Librarian	68

To perform this calculation for other programs, follow this procedure:

1. At the Main Command Level, press [V] to invoke the **Version** command.
2. Write down the `Total Available Memory` as given on the screen.
3. Press [Return] twice to return to the Main Command Level, and use the **Permanent** command to permanently load the program.
4. Press [V] to invoke the **Version** command again.
5. Subtract the current `Total Available Memory` from the value you wrote down in step 2. This is the number of bytes consumed by permanently loading the program. To convert this number to K bytes, divide by 1 024.

[1] K bytes = 1 024 bytes, sizes given are approximate.

Volume Contents

The lists below show how the subsystems, utility programs and libraries are distributed among the Pascal Workstation volumes.

Single-Sided Media

BOOT: Volume Files

```
SYSTEM_P    INITLIB     TABLE       STARTUP     SWVOL.CODE  AUTOSTART
```

BOOT2: Volume Files (Model 320)

```
SYSTEM_P    INITLIB     TABLE       STARTUP     SWVOL.CODE  AUTOSTART
```

SYSVOL: Volume Files

```
LIBRARY
```

ACCESS: Volume Files

```
FILER           EDITOR          LIBRARIAN       MEDIAINIT.CODE TAPEBKUP.CODE
```

CONFIG: Volume Files

```
CTABLE.TEXT     CTABLE3.2.CODE DISC_INTF       DATA_COMM       GPIO
RS232           SRM            F9885           BUBBLE          EPROMS
INTERFACE       EDRIVER        SEGMENTER       HPHIL           MOUSE
DGL_ABS         DGL_REL        ETU.CODE        SYSBOOT
```

LIB: Volume Files

```
IO      LAN     VMELIBRARY
```

GRAPH: Volume Files

```
GRAPHICS
```

CMP: Volume Files

```
COMPILER
```

CMP20: Volume Files

COMPILE20

ASM: Volume Files

ASSEMBLER DEBUGGER REVASM

FLTLIB: Volume Files

FGRAPHICS

FLT20: Volume Files

FGRAPH20

HFS1: Volume Files

HFS_DAM MKHFS.CODE

HFS2: Volume Files

HFSCK.CODE HFS_USER.TEXT OSINSTALL.CODE

HFS3: Volume Files

BACKUP.CODE

DOC: Volume Files

BINDOC.CODE	BINDOC.TEXT	CX.CODE	CX.TEXT	CX0.CODE
CX0.TEXT	CX2.CODE	CX2.TEXT	CXMODULE.CODE	CXMODULE.TEXT
ASMB_M1.CODE	ASMB_M1.TEXT	ASMB_M2.CODE	ASMB_M2.TEXT	ASMB_P1.CODE
ASMB_P1.TEXT	ASMB_P2.CODE	ASMB_P2.TEXT	MOD_2.CODE	MOD_2.TEXT
MOD_3.CODE	MOD_3.TEXT	PROG_1.CODE	PROG_1.TEXT	DEBUG.CODE
DEBUG.TEXT	SYMBOLS.TEXT	BEEP1.ASC	BEEPER1.ASC	BEEPER2.ASC
CLOCK1.ASC	CLOCK2.ASC	CLOCK3.ASC	CRT1.ASC	CRT2.ASC
CRT3.ASC	CRT4.ASC	CRT5.ASC	CRT7.ASC	CRT8.ASC
CRT9.ASC	CRT10.ASC	CRT11.ASC	KBD1.ASC	KBD2.ASC
KBD3.ASC	KBD4.ASC	KBD5.ASC	KBD6.ASC	KBD7.ASC
KBD8.ASC	KBD9P.ASC	KNOB1.ASC	KNOB2.ASC	MASK1.ASC
TIMER1.ASC	TIMER2.ASC	TIMER3.ASC	TIMER4P.ASC	KBD3.CODE
KBD7.CODE				

DGLPRG: Volume Files

DataPoint.TEXT	SinLine.TEXT	SinWindow.TEXT	SinAspect.TEXT	SinViewpt.TEXT
SinLabel1.TEXT	SinLabel2.TEXT	SinLabel3.TEXT	SinAxes1.TEXT	SinClip.TEXT
SinAxes2.TEXT	ConvVtoW.TEXT	ConvWtoV.TEXT	CharCell.TEXT	CsizeProg.TEXT
LdirProg.TEXT	JustProg.TEXT	DrawMdPrg.TEXT	IsoProg.TEXT	AxesGrid.TEXT
LogPlot.TEXT	GstorProg.TEXT	PLineProg.TEXT	PolyProg.TEXT	FillProg.TEXT
FillGraph.TEXT	MarkrProg.TEXT	BAR_KNOB.ASC	BAR_KNOB2.ASC	LOCATOR.ASC
COLOR.ASC				

Double-Sided Media

BOOT: Volume Files

```
SYSTEM_P    INITLIB    TABLE      STARTUP    SWVOL.CODE  AUTOSTART
```

BOOT2: Volume Files

```
SYSTEM_P    INITLIB    TABLE      STARTUP    SWVOL.CODE  AUTOSTART
```

SYSVOL: Volume Files

```
IO          GRAPHICS   LIBRARY
```

ACCESS: Volume Files

```
FILER          EDITOR         LIBRARIAN      MEDIAINIT.CODE TAPEBKUP.CODE
ETU.CODE       CTABLE.TEXT    CTABLE3.2.CODE DISC_INTF      DATA_COMM
GPIO           RS232          SRM            F9885          BUBBLE
EPROMS         INTERFACE      EDRIVER        SEGMENTER      HPHIL
MOUSE          DGL_ABS        DGL_REL        WS1.0_DAM      LAN
SYSBOOT        VMELIBRARY
```

CMP: Volume Files

```
COMPILER       COMPILE20
```

ASM: Volume Files

```
ASSEMBLER      DEBUGGER       REVASM
```

FLTLIB: Volume Files

```
FGRAPHICS      FGRAPH20
```

HFS: Volume Files

```
HFS_DAM        MKHFS.CODE     HFSCK.CODE     OSINSTALL.CODE
BACKUP.CODE    HFS_USER.TEXT
```

DOC: Volume Files

BINDOC.CODE	BINDOC.TEXT	CX.CODE	CX.TEXT	CX0.CODE
CX0.TEXT	CX2.CODE	CX2.TEXT	CXMODULE.CODE	CXMODULE.TEXT
ASMB_M1.CODE	ASMB_M1.TEXT	ASMB_M2.CODE	ASMB_M2.TEXT	ASMB_P1.CODE
ASMB_P1.TEXT	ASMB_P2.CODE	ASMB_P2.TEXT	MOD_2.CODE	MOD_2.TEXT
MOD_3.CODE	MOD_3.TEXT	PROG_1.CODE	PROG_1.TEXT	DEBUG.CODE
DEBUG.TEXT	SYMBOLS.TEXT	BEEP1.ASC	BEEPER1.ASC	BEEPER2.ASC
CLOCK1.ASC	CLOCK2.ASC	CLOCK3.ASC	CRT1.ASC	CRT2.ASC
CRT3.ASC	CRT4.ASC	CRT5.ASC	CRT7.ASC	CRT8.ASC
CRT9.ASC	CRT10.ASC	CRT11.ASC	KBD1.ASC	KBD2.ASC
KBD3.ASC	KBD4.ASC	KBD5.ASC	KBD6.ASC	KBD7.ASC
KBD8.ASC	KBD9P.ASC	KNOB1.ASC	KNOB2.ASC	MASK1.ASC
TIMER1.ASC	TIMER2.ASC	TIMER3.ASC	TIMER4P.ASC	KBD3.CODE
KBD7.CODE	DataPoint.TEXT	SinLine.TEXT	SinWindow.TEXT	SinAspect.TEXT
SinViewpt.TEXT	SinLabel1.TEXT	SinLabel2.TEXT	SinLabel3.TEXT	SinAxes1.TEXT
SinClip.TEXT	SinAxes2.TEXT	ConvVtoW.TEXT	ConvWtoV.TEXT	CharCell.TEXT
CsizeProg.TEXT	LdirProg.TEXT	JustProg.TEXT	DrawMdPrg.TEXT	IsoProg.TEXT
AxesGrid.TEXT	LogPlot.TEXT	GstorProg.TEXT	PLineProg.TEXT	PolyProg.TEXT
FillProg.TEXT	FillGraph.TEXT	MarkrProg.TEXT	BAR_KNOB.ASC	BAR_KNOB2.ASC
LOCATOR.ASC	COLOR.ASC			

Notes:

Pascal Workstation Use on SRM D

The Shared Resource Management system (SRM) connects workstations (computers) to form a network that allows sharing of files and peripherals. This network is controlled by a server.

The Pascal Workstation incorporates features to access shared resources. This appendix outlines the use of a Pascal Workstation on an SRM system.

Note

This appendix summarizes Pascal language features that support the use of SRM and does not include a comprehensive discussion of Pascal language system terms and concepts.

System Concepts

How Your Workstation Connects to SRM

To use the SRM system, you need to know how to move information from one point to another in the system. Everything in the SRM system has a name and/or address. This section is a guided tour of the SRM configuration.

Your Workstation's Identification

Your workstation's contact with the SRM system is through an SRM interface card. This card is in one of the input/output slots in the rear of your computer.

Your workstation is uniquely identified to the SRM system by its node address, assigned when the workstation was added to the SRM system. The node address is set in switches on the SRM interface card.

The SRM interface card also has a setting called the select code, which should normally be left at 21. All of the software shipped to you that uses your workstation's select code expects to see 21. For detailed information on the SRM interface card in your workstation, see the *Shared Resource Management Hardware Installation Manual* or the *Installation Note* that came with the card.

Note that the HP 98643 and the built-in LAN interfaces also are shipped with a default select code of 21. If your system requires both the LAN and SRM interfaces, then you will need to change the select code of one of them. If an SRM is the boot device, the select code can be changed without altering the TABLE program or any other system code. If the SRM is not the boot device, then the TABLE program will have to be changed to accomodate the new select code.

The Connection to the SRM Server

Communication between your workstation and the SRM server can be accomplished by two methods: the coax link configuration and the multiplexer link configuration.

With the coax link, your workstation's SRM interface card is attached to coaxial cable. The coaxial cable connects the server (or servers) and workstations in a "bus" configuration.

With the multiplexer link, your workstation's SRM interface card connects to a multiplexer by a cable. Other workstations may connect to the same multiplexer, with each workstation having a different node address.

The multiplexer has a cable connecting it to the SRM server. The server end of the cable attaches to another SRM interface card in the server.

The Server's Identification

Each server is uniquely identified to the SRM system by its node address, assigned when the server was added to the SRM system.

The node address setting is normally 0 on the SRM interface card(s) in the first SRM server, 1 on the SRM interface card(s) in the second server, and so on.

An SRM system server may contain several SRM interface cards. Unique select code settings distinguish one SRM interface card in the server from another.

Identification of Shared Disc(s)

Each SRM server can have multiple shared disc drives connected to it. From your workstation's point of view (as shipped), only one shared disc is available. The Pascal Workstation automatically configures to have Pascal volume #5: assigned to the SRM system disc.

Additional discs on the SRM system are recognized only if your Pascal CTABLE program is modified. (The procedure is discussed in the *Shared Resource Management Software Installation Manual*'s chapter on "Workstation Startup with Pascal." Typically, your SRM system manager performs this procedure when the disc is configured to the SRM system.)

Unit Numbers for Shared Disc(s)

A workstation connected to an SRM normally has units #5: and #45: set up for SRM shared disc access. The use of two units is in keeping with the idea that there are usually two special volumes (the system volume and the default volume) through which most file accesses occur.

If the workstation is booted from SRM, unit #45: will automatically be configured to be the system volume and unit number #5: will be available for use as the default volume. If there is local mass storage, the system volume can be any volume you desire. To set the system and default volumes, use the What command from the Main Command Level.

Here is how the Filer's Volumes display might look right after booting up a workstation connected to the SRM and having no local mass storage:

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  5 # SRM:
  6   PRINTER:
 45 * SYSTEM42:
Prefix is - SRM:
```

You can see that the system starts out with #5: as the default volume (**Prefix is**) and #45: as the system volume.

The names **SRM**: and **SYSTEM42**: are names of directories in the SRM's hierarchical directory structure (discussed later in this section). In this example, the name of the SRM volume is **SRM**, the workstation we are using is at node address 42, and a directory called *SYSTEM42* exists. **SYSTEM42**: is thus selected as the system volume (denoted by the * beside the directory's name). All of this selecting is done by the Pascal TABLE program as it configures the system each time you boot.

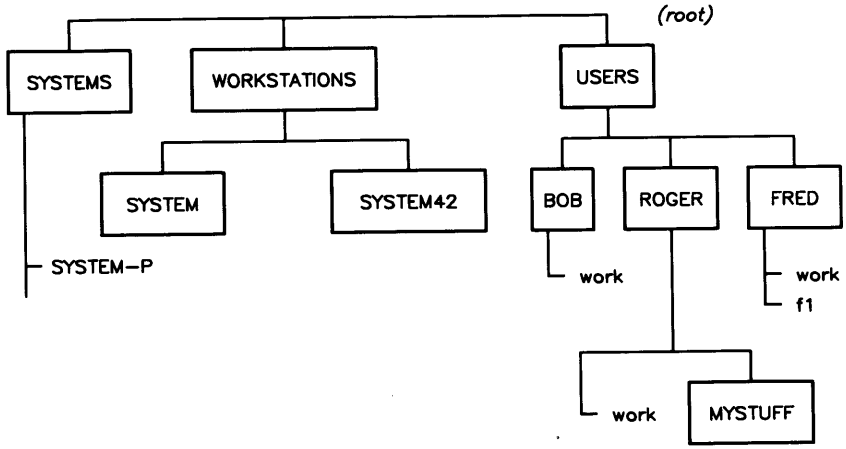
Identification of Shared Peripherals

Shared peripherals (printers and plotters) are connected to the SRM system server. To manage the use of the shared printers and plotters, the server maintains special directories called "spooler directories."

You send information, in files, to a peripheral by placing the file name in the proper spooler directory. The SRM server keeps the files sent to each spooler directory in order and sends the files to the appropriate output device when it becomes available. Once you have placed a file in a spooler directory, your workstation is free to do other processing.

SRM's Hierarchical Directory Structure

The SRM system uses a hierarchical directory structure to organize its files. This directory structure looks like an inverted tree (see example below). The top level in the structure is called the "root." Within the root are directories or files. Each directory may contain files or other directories. The drawing below shows a hierarchical directory structure.



The directory *SYSTEMS* is a special directory used by Boot ROMs version 3.0 or later to automatically load operating or language systems.

Note that within the directory *USERS*, each of the three subordinate directories — *ROGER, BOB* and *FRED* — contains a file called *work*. Because each file and directory is uniquely specified by the list of directories from the root to the file, several files of the same name can exist in the directory structure (provided they are not within the same directory).

Notation

In specifying a location within the directory structure, the root is designated by a slash (/). To refer to a file or directory immediately under the root, for instance the directory *USERS* in the previous illustration, you would write:

```
/USERS
```

To specify a level further down the hierarchy, for instance, the directory *ROGER* under *USERS*, you would write:

```
/USERS/ROGER
```

and for yet another level:

```
/USERS/ROGER/work
```

As you can see, to specify a file, the list of directories to the target file must be specified. Directories in the list are separated by slashes (/). The sequence of names and slash delimiters is called a "directory path," because it indicates the path that leads down the hierarchy to a particular file or directory.

SRM Access Rights

To control access to a file or directory, you can associate passwords with "attributes" that designate the type of access to be protected. Passwords may be assigned when the file or directory is created or with the Filer's **Access** command.

Any access right to which no password is assigned is public. Any user on the SRM may perform operations requiring the access right that is public, whereas only users supplying a correct password can perform operations that require the protected access right.

The attributes and access rights that can be protected with passwords or left public (available to all SRM users) are:

Attribute	Access to File	Access to Directory
READ	operations that read from the file	listing the directory's contents
WRITE	operations that write to the file	adding to or modifying the directory's contents (also need SEARCH to add to directory's contents)
SEARCH	does not apply	accessing the directory's contents
CREATELINK	creating links to the file	creating links to the directory
PURGELINK	removing or renaming the file	removing or renaming the directory
MANAGER	assigning, removing or changing passwords on the file	assigning, removing or changing passwords on the directory
	when password-protected, gives all other access rights as well	when password-protected, gives all other access rights as well

For example, SEARCH access is required on all directories along the path to a given file or directory. Suppose the password *search_pass* protects the SEARCH access on the directory *ROGER* (see illustration). To reach the directory *MYSTUFF* from the root, a user would have to include the password in the directory path specifier to *MYSTUFF*, as in:

```
/USERS/ROGER<search_pass>/MYSTUFF
```

If the SEARCH access capability on *ROGER* were public, any user could "pass through" *ROGER* without supplying a password.

SRM access information is shown in the Filer's Extended directory listing under the heading `directory info`. The one-letter identifiers (`R` for READ, `W` for WRITE, `S` for SEARCH, and so forth) indicate the public access rights on the files listed.

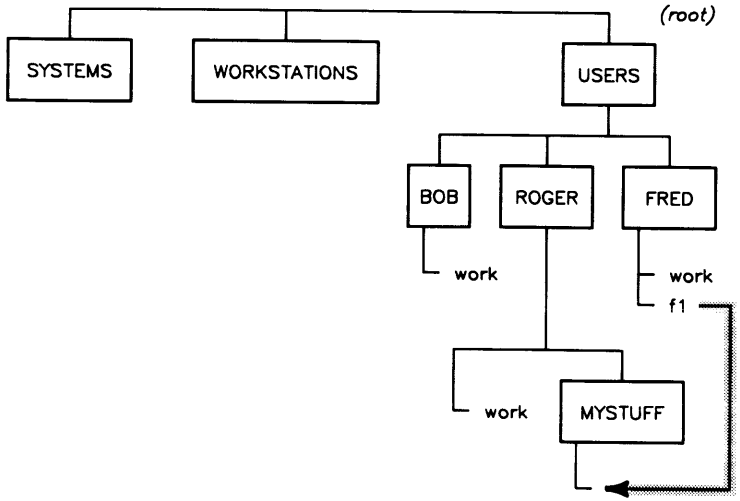
Duplicating Files in More Than One Directory

To save disc space, you can use the Filer's **Duplicate** command to link a file into a directory other than its original location. Once a link has been established, the file looks to the Pascal Workstation as though it is in the directory to which it is linked.

For example, if you created a duplicate link from the file *f1* in the directory *FRED* to the directory *MYSTUFF*, you could specify the path to *f1* as:

```
/USERS/ROGER/MYSTUFF/f1
```

even though no copy of *f1* actually exists in *MYSTUFF*.



This allows you to have access to files without making extra copies of them.

If a file is deleted from a directory to which it was linked, only the directory from which the file was deleted loses access to that file. All other directories with links to the file can still find it. The disc space allocated to a file is only reclaimed when no directories have links to that file.

Using Your Pascal Workstation on SRM

This section describes, through examples, some common procedures you may use to operate your Pascal Workstation on the SRM, including:

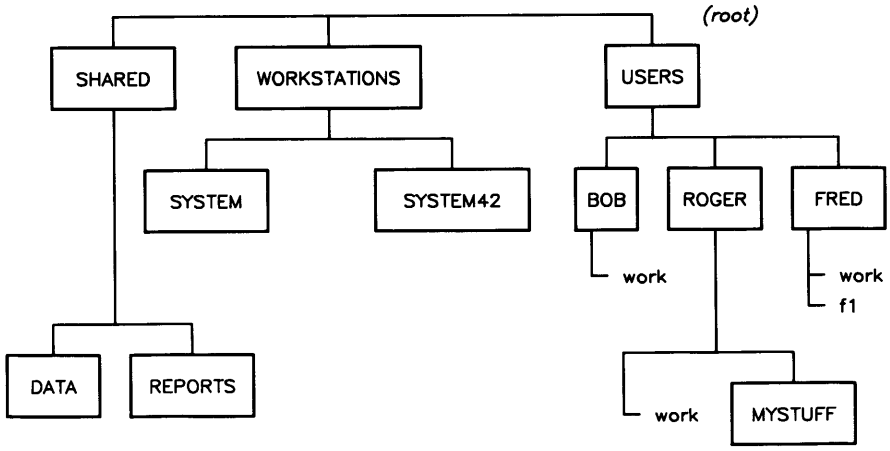
- specifying locations within the hierarchical directory structure;
- creating directories;
- specifying files, directories and volumes;
- protecting access to files and directories by assigning passwords;
- using shared printers and plotters; and
- using Pascal Filer commands with SRM.

Moving Up and Down the Hierarchy

It would be tedious to type a directory path every time you wanted to access a file. To avoid this, you can use the Filer's **Prefix** and **Udir** (unit directory) commands to establish a directory as a reference point from which another location in the directory structure can be specified. A directory path to a file begins either from that reference point or from the root.

To move about within the directory structure, you specify new reference points.

The following discussion uses the directory structure illustrated below:



Using the Prefix Command

Specifying a default volume tells the Pascal file system what volume name to use when none is specified with a file name. With SRM, the Pascal file system considers a directory a "volume." Thus, you may designate a directory as the default volume, and the Pascal file system will:

- perform file accesses directly within that directory or
- use the directory as the beginning of a path to another location in the directory structure.

All volumes recognized by your Pascal Workstation are listed when you use the Filer's **Volumes** command. An example Volumes display for the directory structure illustrated above is:

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  5 # ROGER:
  6   PRINTER:
 45 * SYSTEM42:
 46 # SHARED:
Prefix is - ROGER:
```

In this Volumes listing, the directory *ROGER* is the current default volume. You may designate as the default volume any of the names that are shown in a Volumes listing with the # or * symbol beside them. Because these "volumes" are SRM directory names, you may also designate as the default volume any directory whose location can be specified through one of the names shown.

For example, to designate the directory *MYSTUFF* as the default volume, you would:

1. Press P

   The Filer prompts:

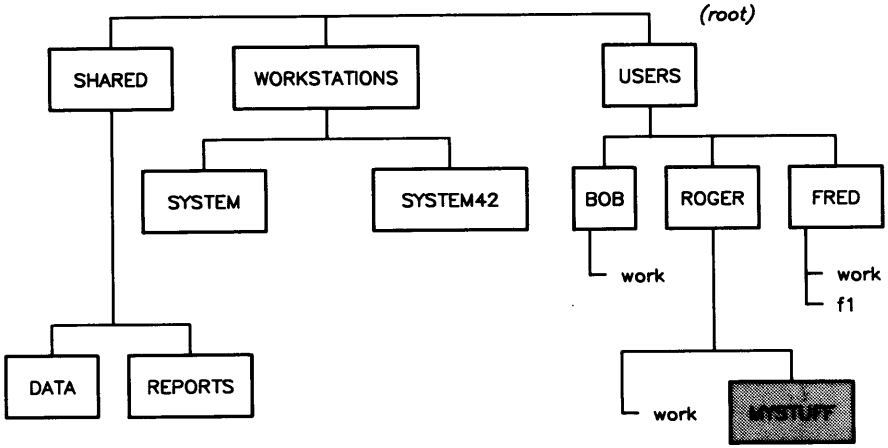
   ```
   Prefix to what directory?
   ```

2. Type

   MYSTUFF Return

With *ROGER:* as the current default volume, the system would assume the path began at *ROGER*. The new Volumes listing (illustrated below) would show *MYSTUFF:* as the default volume and unit #5 would now be associated with *MYSTUFF:*.

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  5 # MYSTUFF:
  6   PRINTER:
 45 * SYSTEM42:
 46 # SHARED:
Prefix is - MYSTUFF:
```



If you then wanted to list the contents of the directory *MYSTUFF*, you would:

1. Press [L] for the Filer's **Ldir** (list directory) command. The Filer prompts:

   ```
   List what directory?
   ```

2. Type

   : [Return]

   (notation that means "default volume")

Next, if you used the **Prefix** command, answering the `Prefix to what directory?` prompt with:

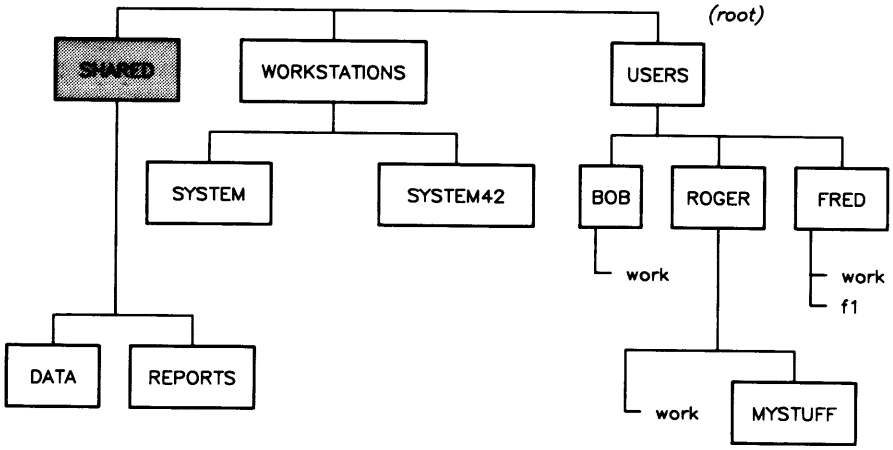
```
#46:  Return
```

or

```
SHARED:  Return
```

your default volume would be at unit #46, not at unit #5.

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  5 # ROGER:
  6   PRINTER:
 45 * SYSTEM42:
 46 @ SHARED:
Prefix is - SHARED:
```



If you then used the **Prefix** command and responded to the prompt with:

```
#5:/USERS/BOB  Return
```

you would not only set the default volume to *BOB:* but you would also change the volume association for unit #5 from *ROGER:* to *BOB:*.

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  5 # BOB:
  6   PRINTER:
 45 * SYSTEM42:
 46 # SHARED:
Prefix is - BOB:
```

The Unit Directory Command

Another way to change the unit number/volume name association is with the Filer's **Udir** (unit directory) command. This association is referred to as setting a "working directory" for a particular unit number. For example, typing:

U (*The Filer prompts* `Set unit to what directory?`)

`#46:DATA` Return

would make *DATA* a working directory by associating unit #46 with that directory, but would not affect the default volume designation (`Prefix` is still *BOB:*).

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  5 # BOB:
  6   PRINTER:
 45 * SYSTEM42:
 46 # DATA:
Prefix is - BOB:
```

Now, because unit #46 is identified with the directory *DATA*, you would be able to use `#46:` and `DATA:` interchangeably with Filer commands. For example, to list the contents of *DATA*, you could respond to the **Ldir** command's `List what directory?` prompt with either:

`#46:` Return

or

`DATA:` Return

Take care when using the **Udir** command **NOT** to assign the default volume's associated unit number to a name other than that of the current default volume. For example, if you were to type:

U (*The Filer prompts* `Set unit to what directory?`)

```
#5:/USERS/ROGER
```

the resulting Volumes display would show:

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  5 # ROGER:
  6   PRINTER:
 45 * SYSTEM42:
 46 # DATA:
Prefix is - BOB:
```

Even though *BOB:* would still be listed as the default volume, you would not be able use *BOB:* as the default volume, because *BOB:* is no longer an "on-line" volume. For instance, to list the contents of the default volume (directory), you would normally:

1. Press L (*The Filer prompts* `List what directory?`)
2. Type

   : Return

   (*which is the notation indicating the default volume*)

   But because, in this example, the Filer looks for and does not find a volume named `BOB:` associated with one of the unit numbers, the Filer does not list the directory's contents, as requested.

Note

You should **NOT** use the **Udir** command to change the association of #45: from your SYSTEM*nn* directory (the system volume). Doing so causes the system to lose access to the system files because the Pascal Workstation looks for a workstation's system files in its SYSTEM*nn* directory.

You may, however, assign any of the on-line volumes as the default volume without affecting the system's access to the system files.

Moving Up the Hierarchical Directory Structure

A special directory name is provided for moving up the hierarchy. Two periods (..) can be used to denote the directory containing the current directory. For instance, if *MYSTUFF:* were the default volume, you could make *ROGER:* the default volume by answering the **Prefix** command's `Prefix to what directory?` prompt with:

.. Return

To go up two levels, use the double-period twice, separated by a slash. For example, to change the default volume from *MYSTUFF:* to *USERS:*, you would respond to the **Prefix** command's prompt with:

../.. Return

The ".." notation can be used to move all the way up the directory structure to the root, although, if you want to go directly to the root, using the slash (/) root identifier is easier.

Creating Directories

SRM directories are created by the Filer's **Make** command. For example, to create a directory called *KAREN* in *USERS* you would:

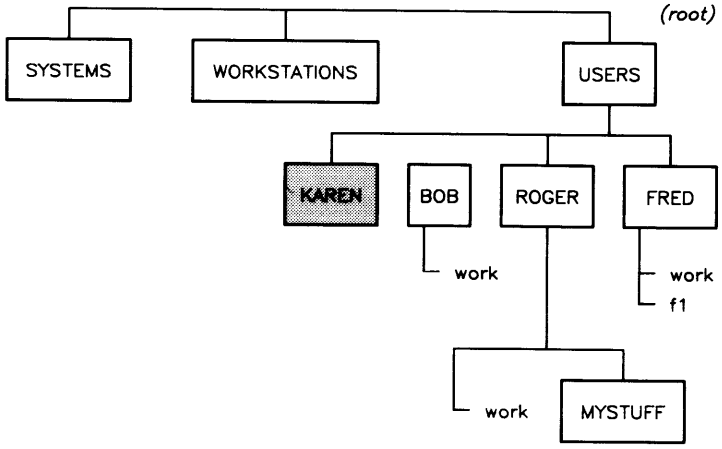
1. Press [M]

   The Filer prompts:

   ```
   Make File or Directory ? (F/D)
   ```

2. Press [D] and specify where you want the directory located:

   ```
   #5:/USERS/KAREN
   ```
   [Return]



Specifying Files, Directories and Volumes

Syntax of File or Directory Specification

The syntax of a legal *file_specification* is given by:

file_specification::=[volume_id] [directory_path] file_name

::= volume_id

In this notation, items between "[" and "]" are optional and quoted items appear literally. Note that, because a directory is a type of file, ***specification for directories is the same as that for files.***

A *file_specification* may appear in one of two forms. The first form consists of an optional *volume_id* followed by a colon (:), then an optional *directory_path*, then a *file_name* which is not optional, then an optional *size_spec.*

An example of the first form is:

```
#45:SYSTEMS/FILER
```

The second form consists of a *volume_id* only. An example of the second form is:

```
#5:
```

Syntax of a Volume Identifier

The volume identifier (*volume_id*) selects one of up to 50 logical units known to the Pascal file system. If no *volume_id* is present, the volume used is the default volume (selected by the Filer's **Prefix** command). Otherwise, the volume is specified in one of three ways:

volume_id::="#"*integer[password]*":"

::= ":"

::=*name[password]*":"

In the first case, *integer* is a one- or two-digit number from 1 to 50. For example, `#23:` is a valid volume identifier. The second case is a special form denoting the default volume. In the third case, *name* is a sequence of characters.

If the volume *name* of the SRM disc is *DISC_ONE*, the disc could be specified as:

```
DISC_ONE:
```

For a logical unit connected to an SRM system, the *volume_id* takes a special meaning. The notation #5: refers to the working directory of unit number five. The notation `#5:/` refers to the root of the SRM directory structure, with which unit number five is associated. The working directory for any SRM volume is selected by the Filer's **Prefix** or **Udir** commands (refer to the "Moving Up and Down the Hierarchy" section earlier in this appendix), or the **What** command of the Main Command Level.

Passwords

Passwords are sequences of up to 16 characters that govern the access rights to a file, a directory or the SRM volume (shared disc). Passwords are assigned to a file or directory either when it is created or by the Filer's **Access** command. The SRM volume password is assigned through the SRM Operating System's RENAME or INITIALIZE commands (refer to the *Shared Resource Management System Manager's Guide* for details).

Including the SRM disc's volume password in a *file_specification* gives you unlimited access to all directories and files on the shared disc. The volume password overrides all other passwords in the system. Because of its power, this password is usually protected and used only with proper authority.

You may use either of the forms for the volume password illustrated in the examples below:

```
#5<volpassword>:/USERS/ROGER/work
#5:<volpassword>/USERS/ROGER/work
```

That is, the volume password may either immediately precede or follow the colon separator.

Syntax of a Directory Path

Directory paths are allowed in Pascal *file_specification* only when specifying files on SRM logical units. The syntax for a *directory_path* is:

directory_path::=["/"] {directory_name [password]"/"}

password::="<" word ">"

directory_name::= file_name

::= "."

::= ".."

The information between "{" and "}" may occur zero or more times. As you can see, there are two special directory names allowed with the SRM. The name "." (a single period) refers to the current working directory. The name ".." refers to the directory containing the working directory. Other names in a *directory_path* are directories along the path to the file or directory being specified.

Note that a *directory_path* does not appear by itself. It appears as part of a *file_specification*, with the *file_name* following the *directory path.* Examples of directory paths are:

`/.<PASS1>/`	*(denotes root, using password PASS1)*
`/USERS/ROGER/`	*(denotes directory ROGER in USERS, where USERS is at the root)*

A specifier including both the *directory_path* and *volume_id* might appear as follows:

```
#5:/WORKSTATIONS/SYSTEM13
```

SRM File or Directory Names

The SRM system allows almost any file name. The Pascal system removes blanks and control characters from file names.

The Pascal SRM Directory Access Method interprets the " < " character as the beginning of a password. All characters up to the next " > " character constitute the password.

Allowable File or Directory Names

What file names are allowed depends on the type of directory used on the volume in which the file resides. In other words, the directory organization determines the file name rules.

File or directory names can consist of alphabetic letters and digits and the the hyphen (-), underscore (_), and period (.) characters. Blanks are removed from file names.

In SRM directories, uppercase and lowercase letters are distinct. (ROGER is not the same as Roger.) You should not use the " / ", " < " and " > " characters, which have special meaning to the SRM system, in file or directory names.

Protecting Access to Files and Directories

The Filer's **Access** command allows you to change public access rights on your SRM files and directories.

To use the **Access** command,

1. Press [A]. The Filer prompts:

   ```
   Access rights for what file?
   ```

2. Type the file or directory specification. For example:

   ```
   #5:/USERS/ROGER/work
   ```
   [Return]

   If the file's MANAGER access is password-protected, you must include the password in the file/directory specification.

   The Filer then prompts:

   ```
   Access: List, Make, Remove, Attributes, Quit?
   ```

3. You may then `List` the existing password(s) and the attributes to which each is assigned, `Make` new password/attribute assignments or `Remove` passwords. `Attributes` is a help feature that lists the attributes options. `Quit` returns you to the Filer prompt.

The attribute option ALL is a shortcut notation for assigning a password to protect all access rights to a file or directory. For example, to assign the password *all_rights* to protect all access rights on the file *work* in the directory *ROGER* (assume all access rights on the directories in the path to *work* and on *work* itself are public), you would:

a. Press [M]. The Filer prompts:

```
Make password:attribute?
```

b. Type:

```
all_rights:ALL [Return]
```

Now, any user wishing to access *work* for operations requiring any of the access rights (READ, WRITE, MANAGER, SEARCH, CREATELINK, PURGELINK) must include the password in the file specifier, for example:

```
#5:/USERS/ROGER/work<all_rights>
```

Using Shared Printers and Plotters

To output data on a shared printer or plotter, you must place your data in a file in the printer's or plotter's spooler directory. Once the file is in the directory, the SRM Operating System sends the file to the appropriate output device as soon as the device is free.

For example, to print the text file named *JOB_1.TEXT* located by the SRM directory path *#5:/PROJECT_1* on the printer assigned to spooler directory named *LP*:

1. Press [F] to enter the Filer from the Main Command Level.
2. Press [T] to invoke the Filer's **Translate** command.

   The Filer prompts:

   ```
   Translate what file?
   ```

3. Type:

   ```
   #5:/PROJECT_1/JOB_1.TEXT,#5:/LP/JOB_1.ASC [Return]
   ```

The file will be printed as soon as the printer is available. The `.ASC` ending on the file name tells the Filer to translate the information file into ASCII format, which is best handled by the SRM and its supported peripherals.

In contrast to the SRM 1.0 system, SRM 2.0 and later systems allow non-ASCII files to be sent to the printing device as a byte stream.

Note

Any non-ASCII (such as .TEXT) file sent to the spooler is printed exactly as the byte stream sent. Unless you set up your non-ASCII file correctly, improper printer output or operation could result. Therefore, it is recommended that you use only ASCII type files when spooling to a printer or plotter.

Using Pascal Filer Commands With SRM

The following table summarizes the Filer commands either used exclusively with SRM or whose use has a special meaning for SRM.

Command	Significance to SRM
Access	Changes the access rights (passwords) on a file or directory
Bad_secs	Not valid for SRM
Duplicate	Links a file or files to a directory at another location in the directory structure
Hfs	Not valid for SRM
Krunch	Not valid for SRM
Make	Creates a directory within the SRM directory structure
Prefix	Sets or changes the default volume name
Translate	Translates a file into ASCII format for output on a shared printer or plotter
Udir	Changes the path specification for a unit directory
Zero	Not valid for SRM

The Hierarchical File System

E

The Hierarchical File System (HFS) is the fourth HP file system supported by the Pascal Workstation. Since HFS is already used by the Series 300 HP-UX operating system and also supported by the BASIC Workstation, it is now possible for all three systems to boot from and share files on a single disc.

Unlike Logical Interchange Format (LIF) discs, where all files appear in one directory, HFS discs allows sub-directories (i.e. directories to be "nested" within other directories). The HFS hierarchy is similar to the one used by the Shared Resource Manager (SRM) file system.

This appendix describes the following HFS operations.

- HFS disc initialization (using MEDIAINIT for HFS)
- Making the HFS directory structure (the MKHFS utility)
- HFS changes to the Filer (including the Hfs command)
- Checking the integrity of an HFS disc (the HFSCK utility)
- Copying Workstation files to an HFS disc
- Installing system boot files on HFS discs (the OSINSTALL utility)

Further information regarding HFS can be found in the *Workstation System Manual.* If you are planning to install the Pascal Workstation on an existing HP-UX Workstation disc, refer to the procedures in the *Workstation System Manual.*

Typically, any Winchester (hard) disc supported by the Pascal Workstation can be used as an HFS disc. Although flexible discs can also be used as HFS discs, there will be less usable space on an HFS flexible disc than on a LIF flexible disc.

Note

This appendix summarizes Pascal Workstation features that support the use of HFS discs and does not include a comprehensive discussion of the Pascal system terms and concepts.

HFS Basics

Before looking at the HFS directory structure, it is useful to know what is necessary to access HFS discs, to boot the workstation from an HFS disc, and how an HFS disc appears to the Pascal Workstation

HFS Modules

This appendix assumes that the HFS_DAM (HFS Directory Access Method) module has been installed in the Pascal Workstation. There are two ways to install the module; it can be included in the initial library (`INITLIB`) or it can be executed from the keyboard as follows:

1. Press [X] and type: `HFS:HFS_DAM.` (or `HFS1:HFS_DAM.`) [Return]
2. Press [X] and type: `BOOT:TABLE.` (or `BOOT2:TABLE.`) [Return]

The first step installs the module from the HFS: disc (or the HFS1: disc if your workstation software was delivered on single-sided discs.) The second step executes the TABLE program to configure the workstation. TABLE is found on the BOOT: disc (use BOOT2: disc if you are using bit-map display hardware).

Booting from an HFS Disc

If you wish to boot the Pascal Workstation from an HFS disc, you need to create a special directory, copy all the boot files to the HFS disc, and install the SYSTEM_P boot file with a special utility. All of this is explained later in this appendix.

HFS Disc Unit Numbers

An HFS Winchester disc normally has unit #11: set up for access. If the workstation is booted from an HFS hard disc, #46: will automatically be configured to be the system volume and unit number #11: will be the default volume. To reset the system and default volumes, use the What command from the Main Command Level.

Here is how the Filer's Volumes display might look right after booting up a workstation from an HFS disc.

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  6   PRINTER:
 11 # HDISC:
 46 * SYSTEM:
Prefix is - HDISC:
```

The names HDISC: and SYSTEM: are names of directories in the hierarchical directory structure. In this example, the name of the default volume is HDISC, and the name of the system volume is SYSTEM (denoted by the * beside the directory's name). All of this selecting is done by the Pascal TABLE program as it configures the system each time you boot.

For flexible disc drives, a LIF disc is typically accessed by unit #3: or unit #4:. To access an HFS flexible disc in one of these units, the Pascal TABLE program sets up unit #43: and #44:. Thus, two unit numbers (i.e. #3 and #43) are assigned to the same physical disc drive. Since an HFS disc has a very small LIF directory header at the beginning of the disc, an HFS flexible disc will appear twice in the Volumes listing, once as a LIF disc and again as an HFS disc. **Remember that the LIF volume must not be used on an HFS disc.** It is there for installing SYSTEM files only. In fact, if the LIF volume of a hard disc appears instead of the HFS volume, in a volumes listing, you may not have installed the HFS_DAM module described above.

CAUTION

Possible loss of data. Every HFS disc has a small LIF "boot area" at the beginning of the disc. You should never attempt to save a file to the LIF portion of an HFS disc. Doing so will destroy files on the HFS disc.

Only utilities provided with the Pascal Workstation, such as the OSINSTALL command, should be used to install files in the boot area of an HFS disc.

HFS Disc Initialization

All discs must be initialized before they can be used by the computer. Both LIF and HFS discs are initialized by using the **MEDIAINIT.CODE** utility. MEDIAINIT not only initializes the disc, but also creates a LIF directory structure on the disc. To make the disc HFS compatible, you must execute the **MKHFS.CODE** utility after running MEDIAINIT. But first, note these differences when initializing a disc for HFS use.

To initialize a disc for HFS usage:

1. From the Main Command Level, press [X], type **ACCESS:MEDIAINIT** and press [Return].

   The MEDIAINIT display will appear something like this:

```
Mediainit [Rev 3.2 1/15/87] 15-Feb-87 09:03:07

     Copyright 1987 Hewlett-Packard Company.
              All rights reserved.

Volume ID?
```

2. Type the unit number of the disc drive you wish to initialize. Typically, a Winchester disc will be assigned to unit #11.

   For practice, you may wish to create an HFS flexible disc. Find a new unused flexible disc and insert it in unit #3.

   For this practice example, type: **#3:** [Return]

3. MEDIAINIT will examine the disc and print the results.

```
Mediainit [Rev 3.2 1/15/87] 15-Feb-87 09:03:07

     Copyright 1987 Hewlett-Packard Company.
             All rights reserved.

Volume ID? #3:

Device HP9122 removable disc, 700, 0, 0
Logical unit #3 - <no directory>
WARNING: the initialization will also destroy:
 #43: <no dir>

Are you SURE you want to proceed? (Y/N)
```

MEDIAINIT is telling you that the disc has no directory. If MEDIAINIT prints the name of a directory, then the disc has already been initialized and may contain valuable information. Be sure you do not want what is already on the disc since once the disc has been initialized, all previous information on the disc will be lost.

If you are satisfied that you can use the disc, press [Y] to begin initialization.

4. When initializing a disc in a drive that has more than one formatting option, the following line will appear.

   ```
   FORMATTING option? (defaults to 0)
   ```

   The **formatting option**, which indicates the block size for the disc, should be set to 3 for HFS double-sided micro discs. Type: 3 [Return].

5. When initializing a disc in a drive that can have more than one interleave factor, the following line will appear.

   ```
   Interleave factor? (defaults to 2)
   ```

   The **interleave factor** recommended by MEDIAINIT should provide optimum performance. Accept the default value by pressing [Return].

6. The disc is now being initialized. The operation usually takes a few minutes (but may take longer, sometimes more than an hour for very large Winchester discs). When initialization is complete, MEDIAINIT will display:

```
Medium initialization completed
```

7. A LIF directory structure is now being written on the disc. When this is completed, MEDIAINIT will display:

```
Volume zeroing completed
```

8. The media is now ready for use as a LIF disc. The next section shows how to prepare the disc for HFS by using the MKHFS utility.

Making an HFS Disc

The MKHFS utility creates an HFS directory structure on-top-of an existing LIF disc. The utility is included on the HFS: disc (on the HFS1: disc if your workstation system software files were ordered on single-sided media).

Just like MEDIAINIT, MKHFS should only be executed once for a disc; it will destroy all existing information on the disc. Execute the utility as follows:

1. From the Main Command Level, press [X] and type: **HFS:MKHFS** (or **HFS1:MKHFS**) and press [Return]

   The following display will appear.

```
MKHFS [Rev 3.2 1/15/87]  15-Feb-87 08:20:17

   Copyright 1987 Hewlett-Packard Company.
            All rights reserved.

Volume ID?
```

   If you were initializing an HFS Winchester disc you would type #11, but continuing the flexible disc example started in the last section, you should type **#3**: [Return] as the response.

3. MKHFS checks the disc and asks if you wish to change the default parameters.

```
MKHFS [Rev 3.2 1/15/87]  15-Feb-87 08:20:17

   Copyright 1987 Hewlett-Packard Company.
          All rights reserved.

Volume ID? #3:

Device: HP9122 removable disc, 700, 0, 0
Logical unit #3 - 'V3:'
WARNING: the initialization will also DESTROY:
 #43 <no dir>
Change or examine default parameters? (y/n)
```

CAUTION

With the 3.2 version of the Pascal Workstation System, certain combinations of these parameters may result in corruption of HFS file systems. Therefore, do **NOT** change the default values of these parameters. We plan to fix this problem with a subsequent revision of the system. We apologize in advance for any inconvenience this may have caused.

Type: [N] (for reasons stated in the Caution note above).

4. The utility will now prompt:

```
Are you SURE you want to overwrite the disc? (y/n)
```

If you are satisfied that you can use the disc, press [Y] to create the HFS directory structure. The operation will not take long and you will see various messages displayed. When finished, the Main Command Level prompt will appear. The HFS disc is now ready for use.

You have now created an HFS directory structure on the disc. The only "file" you will find on the disc is the directory: `/lost+found` which is created on each HFS volume. Do not remove or rename this directory as it is necessary to the integrity of the disc.

If you need to convert a Winchester disc back from HFS to LIF format, follow these steps:

1. Execute `ACCESS:MEDIAINIT.CODE` and specify that the Winchester disc be initialized by entering a volume specifier similar to this: `#11`
2. Next execute `BOOT:TABLE`. (or `BOOT2:TABLE`.).
3. Zero each LIF unit on the disk (for example, `#11`, `#12`, etc.) using the FILER.

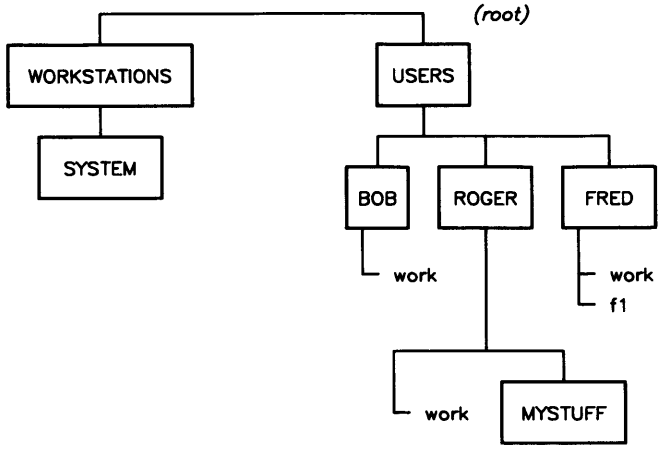
Note that the LIF volumes on the Winchester disc will not be "seen" by the system until TABLE is run (or the system is rebooted), and the LIF volumes zeroed. Steps 2 and 3 are not needed for flexible discs, as unit `#3` (or `#4`, etc.) provides direct access to the LIF volume on the disc.

CAUTION

Step 1 above (executing `MEDIAINIT.CODE`) will erase all files that were on the HFS disc. Make a backup of any important files before beginning the above procedure.

HFS's Hierarchical Directory Structure

HFS uses a hierarchical directory structure to organize its files. This directory structure looks like an inverted tree (see example below). The top level in the structure is called the "root." Within the root are directories or files. Each directory may contain files or other directories. The drawing below shows a hierarchical directory structure.



Typical Hierarchical Directory Structure

Note that within the directory USERS, each of the three subordinate directories — ROGER, BOB and FRED — contains a file called **work**. Because each file and directory is uniquely specified by the list of directories from the root to the file, several files of the same name can exist in the directory structure (provided they are not within the same directory).

Notation

In specifying a location within the directory structure, the root is designated by a slash (/). To refer to a file or directory immediately under the root, for instance the directory USERS in the previous illustration, you would type:

```
/USERS
```

To specify a level further down the hierarchy, for instance, the directory ROGER under USERS, you would type:

```
/USERS/ROGER
```

and for yet another level:

```
/USERS/ROGER/work
```

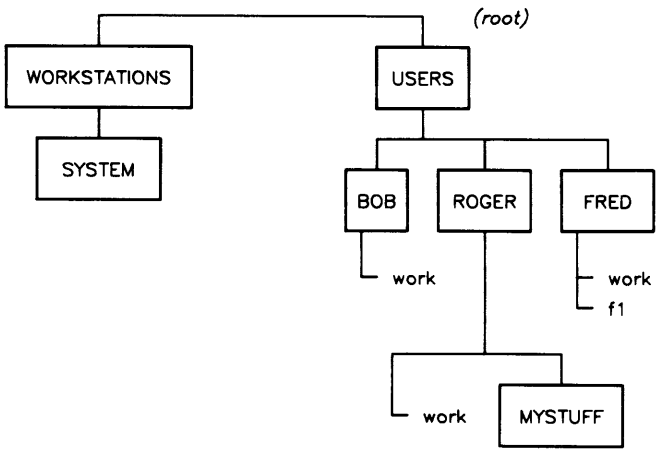
As you can see, to specify a file, the list of directories to the target file must be specified. Directories in the list are separated by slashes (/). The sequence of names and slash delimiters is called a *directory path*, because it indicates the path that leads down the hierarchy to a particular file or directory. If you refer to both the directory path and the file name at the same time, it is often called a *pathname*.

Moving Up and Down the Hierarchy

It would be tedious to type a directory path every time you wanted to access a file. To avoid this, you can use the Filer's **Prefix** and **Udir** (unit directory) commands to establish a directory as a reference point from which other locations in the directory structure can be specified. A directory path to a file begins either from that reference point or from the root.

To move about within the directory structure, you specify new reference points.

The following discussion uses the directory structure illustrated below:



Directory Structure

Using the Prefix Command

Specifying a default volume tells the Pascal file system what volume name to use when none is specified with a file name. With HFS discs, the Pascal file system considers a directory to be a "volume." Thus, you may designate a directory as the default volume, and the Pascal file system will:

- perform file accesses directly within that directory or
- use the directory as the beginning of a path to another location in the directory structure.

All volumes recognized by your Pascal Workstation are listed when you use the Filer's **Volumes** command. An example Volumes display for the directory structure illustrated above is:

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  6   PRINTER:
 11 # ROGER:
 46 * SYSTEM:
Prefix is - ROGER:
```

In this Volumes listing, the directory ROGER is the current default volume. You may designate as the default volume any of the names that are shown in a Volumes listing with the # or * symbol beside them.

You may also choose to set the default volume to be any directory that exists on the HFS disc. For example, to designate the directory MYSTUFF (a directory "under" the directory ROGER) as the default volume, you would:

1. Press P

   The Filer prompts: `Prefix to what directory?`

2. Type: `MYSTUFF` Return

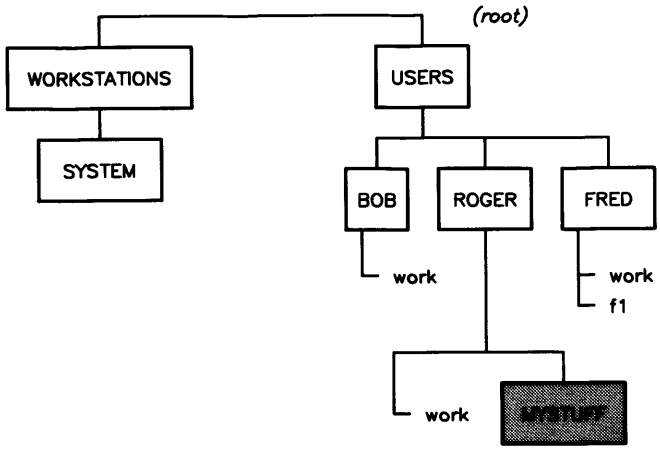
Note

You should **not** use the **Prefix** (or **Udir**) command to change the association of unit **#46**: with your SYSTEM: directory (the system volume). Doing so causes the system to lose access to the system files, because the Pascal Workstation looks for a workstation's system files in its SYSTEM: directory. You may, however, assign any of the on-line volumes to be the default volume without affecting the system's access to the system files.

When Prefixing an HFS-formatted **flexible** disc, you are automatically prefixed to the root directory; you **cannot** prefix to any directory below the root level on this type of flexible disc. (However, with HFS-formatted **hard** discs, this restriction does **not** apply.)

With ROGER: as the current default volume, the system would assume the path began at ROGER. The new Volumes listing (illustrated below) would show MYSTUFF: as the default volume and unit #11 would now be associated with MYSTUFF:.

```
Volumes on-line:
  1    CONSOLE:
  2    SYSTERM:
 11 # MYSTUFF:
  6    PRINTER:
 46 # SYSTEM:
Prefix is - MYSTUFF:
```



Directory Structure for MYSTUFF

If you then wanted to list the contents of the directory MYSTUFF, you would:

1. Press [L] for the Filer's **Ldir** (list directory) command. The Filer prompts:
   `List what directory?`
2. Type:
   : [Return]
   (*notation that means "default volume"*)

Next, if you used the **Prefix** command, answering the `Prefix to what directory?` prompt with:

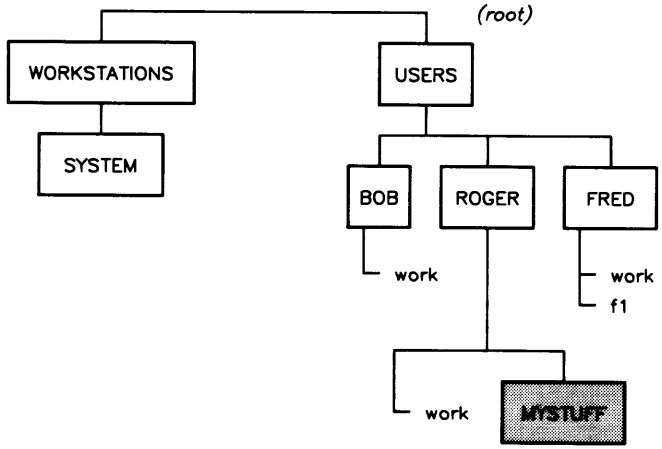
`#46:` [Return]

or

`SYSTEM:` [Return]

your default volume would be at unit #46, not at unit #11.

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
 11 # ROGER:
  6   PRINTER:
 46 * SYSTEM:
Prefix is - SYSTEM:
```



Directory Structure for SYSTEM

If you then used the **Prefix** command and responded to the prompt with:

```
#11:/USERS/BOB    Return
```

you would not only set the default volume to BOB: but you would also change the working directory for unit #11 from ROGER: to BOB:.

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  6   PRINTER:
 11 # BOB:
 46 * SYSTEM:
Prefix is - BOB:
```

The Unit Directory Command

Another way to change the unit number/volume name association is with the Filer's **Udir** (unit directory) command. This association is referred to as setting a "working directory" for a particular unit number. For example, typing:

U (*The Filer prompts* `Set unit to what directory?`)

```
#11:/USERS/BOB/DATA    Return
```

would make DATA a working directory by associating unit #11 with that directory, but would not affect the default volume designation (`Prefix` is still BOB:).

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  6   PRINTER:
 11 # DATA:
 46 * SYSTEM:
Prefix is - BOB:
```

Now, because unit #11 is identified with the directory DATA, you would be able to use `#11:` and `DATA:` interchangeably with Filer commands. For example, to list the contents of DATA, you could respond to the **Ldir** command's `List what directory?` prompt with either:

```
#11:    Return
```

or

```
DATA:    Return
```

Notice that the default volume, BOB:, is no longer an "on-line" volume. For instance, to list the contents of the default volume (directory), you would normally:

1. Press [L] (*The Filer prompts* `List what directory?`)
2. Type

   : [Return]

   (*which is the notation indicating the default volume*)

   But because, in this example, the Filer looks for and does not find a volume named `BOB:` associated with one of the unit numbers, the Filer does not list the directory's contents, as requested.

Note

You should **not** use the **Udir** (or **Prefix**) command to change the association of unit `#46:` with your SYSTEM: directory (the system volume). Doing so causes the system to lose access to the system files, because the Pascal Workstation looks for a workstation's system files in its SYSTEM: directory. You may, however, assign any of the on-line volumes to be the default volume without affecting the system's access to the system files.

When Prefixing an HFS-formatted **flexible** disc, you are automatically prefixed to the root directory; you **cannot** prefix to any directory below the root level on this type of flexible disc. (However, with HFS-formatted **hard** discs, this restriction does **not** apply.)

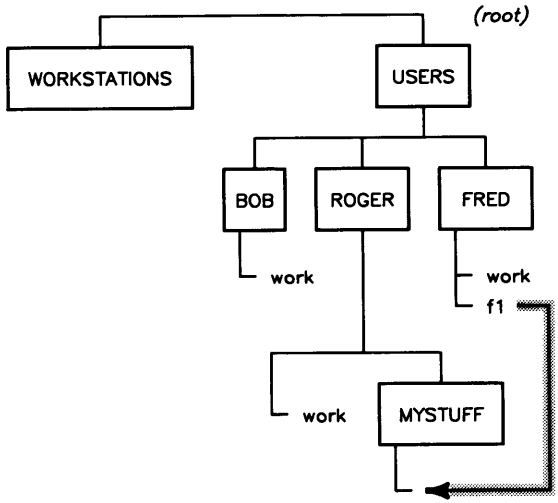
Duplicating Files in More Than One Directory

To save disc space, you can use the Filer's **Duplicate** command to link a file into a directory other than its original location. Once a link has been established, the file looks to the Pascal Workstation as though it is in the directory to which it is linked.

For example, if you created a duplicate link from the file f1 in the directory FRED to the directory MYSTUFF, you could specify the path to f1 as:

```
/USERS/ROGER/MYSTUFF/f1
```

The contents of the file f1 were not actually copied to the new directory (only the name was copied, i.e the link).



Creating Duplicate Files

This allows you to have alternate access to files without making extra copies of them.

If a file is deleted from a directory to which it was linked, only the directory from which the file was deleted loses access to that file. All other directories with links to the file can still find it. The disc space allocated to a file is only reclaimed when no directories have links to that file.

The Filer's **Duplicate** command can also be used to move a file from one HFS directory to another. Since this is done by changing the link to the file and not by moving the file itself, files can be moved very quickly. You cannot **Duplicate** files from one disc to another disc.

Moving Up the Hierarchical Directory Structure

A special directory name is provided for moving up the hierarchy. Two periods (..) can be used to denote the directory containing the current directory. For instance, if MYSTUFF: were the default volume, you could make ROGER: the default volume by answering the **Prefix** command's `Prefix to what directory?` prompt with:

.. Return

To go up two levels, use the double-period twice, separated by a slash. For example, to change the default volume from MYSTUFF: to USERS:, you would respond to the **Prefix** command's prompt with:

../.. Return

The ".." notation can be used to move all the way up the directory structure to the root, although, if you want to go directly to the root, using the slash (/) root identifier is easier.

Creating Directories

Directories are created by the Filer's **Make** command. For example, to create a directory called KAREN in USERS you would:

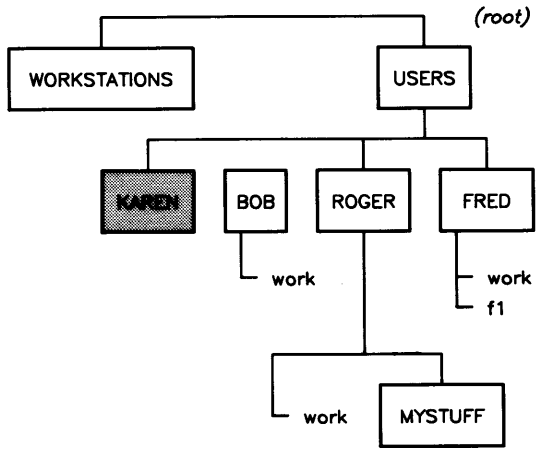
1. Press [M]

   The Filer prompts:

   ```
   Make File or Directory ? (F/D)
   ```

2. Press [D] and specify where you want the directory located:

   ```
   #11:/USERS/KAREN    Return
   ```



Adding to the Directory Structure

Specifying Files, Directories and Volumes

Syntax of File or Directory Specification

The syntax of a legal *file_specification* is given by:

file_specification::=[volume_id] [directory_path] file_name

::= volume_id

In this notation, items between "[" and "]" are optional and quoted items appear literally. Note that, because a directory is a type of file, ***specification for directories is the same as that for files.***

A *file_specification* may appear in one of two forms. The first form consists of an optional *volume_id* followed by a colon (:), then an optional *directory_path*, then a *file_name* which is not optional, then an optional *size_spec*.

An example of the first form is:

```
#46:SYSTEM/FILER
```

The second form consists of a *volume_id* only. An example of the second form is:

```
#11:
```

Syntax of a Volume Identifier

The volume identifier (*volume_id*) selects one of up to 50 logical units known to the Pascal file system. If no *volume_id* is present, the volume used is the default volume (selected by the Filer's **Prefix** command). Otherwise, the volume is specified in one of four ways:

volume_id::="#"integer":"

::= ":"

::= ""*

::=name":"

In the first case, *integer* is a one- or two-digit number from 1 to 50. For example, **#23:** is a valid volume identifier. The second case is a special form denoting the default volume. In the third case, *name* is a sequence of characters.

For a logical unit, the *volume_id* takes a special meaning. The notation #11: refers to the working directory of unit number eleven. The notation **#11:/** refers to the root of the directory structure with which unit number eleven is associated. The working directory for any volume is selected by the Filer's **Prefix** or **Udir** commands (refer to the "Moving Up and Down the Hierarchy" section earlier in this appendix), or the **What** command of the Main Command Level.

Syntax of a Directory Path

Directory paths are allowed in Pascal *file_specification* when specifying files on logical units. The syntax for a *directory_path* is:

directory_path::=["/"] {directory_name "/"}

directory_name::= file_name

::= "."

::= ".."

The information between " { " and " } " may occur zero or more times. As you can see, there are two special directory names allowed with HFS. The name " . " (a single period) refers to the current working directory. The name " .. " refers to the directory containing the working directory. Other names in a *directory_path* are directories along the path to the file or directory being specified.

Note that a *directory_path* does not appear by itself. It appears as part of a *file_specification*, with the *file_name* following the *directory path.* Examples of directory paths are:

/	(*denotes root*)
/USERS/ROGER	(*denotes directory ROGER in USERS, where USERS is at the root*)

A specifier including both the *directory_path* and *volume_id* might appear as follows:

```
#11:/WORKSTATIONS/SYSTEM
```

File or Directory Names

The system allows almost any file name. The Pascal system removes blanks and control characters from file names.

File or directory names can consist of alphabetic letters and digits, the hyphen (-), underscore (_), and period (.) characters. Blanks are removed from file names. Uppercase and lowercase letters are distinct. (ROGER is not the same as Roger.) You should not use the " / " character, which has special meaning to the system, in file or directory names.

Long File Name HFS

Only short file name HFS discs are supported on the Pascal Workstation, unlike HP-UX (version 6.2 and later) which supports both short and long file name HFS discs.

File Mode

Each file has a **file mode** which is a three-digit number indicating who may read or modify the file. In general, the changing the file mode is not necessary for the Pascal Workstation. Complete information about file modes is included in the *Workstation System Manual.*

Note

To execute a Pascal program on the Pascal Workstation, you must have read-permission for the file (execute-permission is not necessary).

The HP-UX Workstation usually requires execute-permission to execute a file.

If you are sharing an HFS disc with an HP-UX workstation, and have an ownership or permission problem, the HP-UX System Administrator ("root-user") can help you.

User Identification

Just as a file has a mode, a user has a user-id. Fortunately, a user-id is just a number, no complex calculations needed. The Pascal Workstation has been assigned a user-id of 17. That is to say, all Pascal users who save files on an HFS disc have their files marked with a user-id of 17. Similar to the user-id there is a group-id assigned to the Pascal Workstation. The group-id is 9 (which is also the group-id of the HP BASIC Workstation). These numbers are shown when you list an HFS directory.

Listing HFS Files

To see the file modes and user identification, use the Filer's **Ext-Dir** command to list an HFS directory. You will see something similar to the following.

```
HDISC:                  Directory type= HFS 755    17    9
created                           block size=1024
changed  15-Feb-87  8:14:19 Storage order

...file name....     # blks     # bytes   start blk ....last change.. extension1
                 type  t-code ..directory info... ....create date.. extension2

PROJECT.TEXT                3        2048           4 15-Feb-87 10.51.33          1
                 Text    -5570  666m     17u      9g                              -1
FILES shown=1 allocated=1
BLOCKS (1024 bytes) used=3 unused=176
```

HFS Extended Listing

The listing consists of 9 lines (excluding blank lines). Several of the lines are explained below.

Line 1: The first line shows the name of the directory (`HDISC:`), the directory type (HFS), the file mode of the directory (755), the user-id (17) and the group-id (9).

Line 2: This line shows the block size for this disc (1024).

Line 6: This is the first of two lines describing the file named (`PROJECT.TEXT`).

Line 7: This is the second of two lines describing the file named (`PROJECT.TEXT`). The line indicates a text-type file (t-code = -5570) the file mode: `666m` (where `m` means mode), the user-id `17u` (where `u` means user), and the group-id `9g` (where the `g` means group).

If you look back at line 7, you will see that the file mode was `666m`. If the file was a directory, the file mode would be `d666m`.

The Hfs Command

The following "walk-through" of the Hfs command will help you become familiar with the file mode, user-id, group-id, and the umask.

To use the **Hfs** command,

1. Press [H]. The Filer prompts:

   ```
   HFS Access: Owner, Group, Mode, Umask, Quit
   ```

2. For Owner, Group, and Mode the Filer prompts:

   ```
   For which file ?
   ```

   Type the file or directory specification. For example:

   ```
   #11:/USERS/ROGER/PROJECT.TEXT    [Return]
   ```

3. For Owner the Filer prompts: **`Enter new owner number.`**

   For Group the Filer prompts: **`Enter new group number.`**

   Unless you are sharing files with a BASIC or HP-UX workstation, you will not need to change the owner-id or the group-id. Changing either may prevent you from accessing your file. For example, if you have set the file mode so that only the owner of a file may read the file, and then change the ownership to some other user-id, you will no longer be able to read the file. If the disc is shared with an HP-UX workstation, the HP-UX System Administrator ("root-user") can change the permissions.

   If you do share files with a BASIC or HP-UX workstation, there will be some other user-id (owner) or group-id to which you may wish to assign the file. Once the owner is changed, only the new owner or the HP-UX "root-user" can return ownership back to you.

Note

Changing the ownership of a file may prevent you from accessing the file.

For the Mode mode command, the Filer prompts: **`Enter new mode.`**

Entering a new mode will change the access permission for the file. For example, if you wish to restrict a file so that only you can read it, you might enter a mode of 400. The Filer's **Ext-dir** command can then be used to see the new file mode.

Umask

The Umask (user-mask) command of the **Hfs** menu determines the **file mode** for all files and directories that you create. Normally, directories are created with 777 mode (all permissions) and files with 666 mode (read and write permission). By changing the Umask, you can specify a new file mode to be assigned when a file is created.

To specify a Umask, you must determine the complement of the file mode. For example, to set 777 as the file mode, you would set the Umask to 000. As another example, if you wanted file mode 640, you would set the Umask to 137. Since the file mode and Umask are complements of each other, their sum should always be 777.

The Umask can be independently set for each HFS unit on the Pascal Workstation. The Umask command will prompt:

```
For which unit ?
```

Enter the unit number of the disc, for example, `#11:`. You are then prompted:

```
Enter new umask number
```

Type the three-digit umask for the unit.

When you finish with the **Hfs** menu, press [Q] to Quit the menu and return to the Filer's command menu.

Summary of Filer Commands for HFS

The following table summarizes the Filer commands either used exclusively by HFS or whose use has a special meaning for HFS discs.

Command	Significance to HFS
Access	Not valid for HFS
Bad_secs	Not valid for HFS
Duplicate	Links a file or files to a directory at another location in the directory structure
Hfs	Sets or changes file ownership and file permission.
Krunch	Not valid for HFS
Make	Creates a directory within the HFS directory structure
Prefix	Sets or changes the default volume name
Translate	Translates a file into ASCII (or other) format.
Udir	Changes the path specification for a unit directory
Zero	Not valid for HFS

Only Filer commands particular to HFS discs are described in this appendix.

Checking HFS Disc Integrity

The HFSCK utility allows you to check the integrity of your HFS disc and repair it if necessary. The HFSCK utility is provided on the HFS: system disc (double-sided media) or the HFS2: system disc (single-sided media).

1. From the Main Command Level, press [X] and then type: `HFS:HFSCK` (or `HFS1:HFSCK`) [Return]
2. The HFSCK utility will display:

```
HFSCK [Rev 3.2 1/15/87] 15-Feb 87 09:01:05

   Copyright 1987 Hewlett-Packard Company.
            All rights reserved.

Volume ID?
```

3. You will want to enter the unit number of your HFS disc. If you are using a Winchester disc, type `#11`: [Return]. If you are using the flexible disc created earlier in this appendix, type `#43`: [Return]. (We use `#43` instead of `#3` for flexible discs to indicate to the system to expect an HFS disc instead of a LIF disc.)
4. The HFSCK utility will now prompt:

   ```
   Report only, Ask always, or Normal confirmation (r/a/n)?
   ```

   In general, you will use [N] for normal confirmation. If you wish to check the disc without changing anything, you can type [R].
5. HFSCK now prompts:

   ```
   Alternate superblock number (RETURN for default)?
   ```

   In general, you can simply press [Return]. The superblock is somewhat like an index to a book. Without it, you may not easily find what you want to see. If somehow the first part of your disc has been scrambled, you can specify an alternate superblock that was displayed when you formatted the disc with the MKHFS command. If you do not remember what these numbers were, and the default does not work, you might retry with 16.
6. In this example we will choose [R] for a Report only. Press [R].

7. The system will take a few moments and report the following (assuming the disc is all right):

```
** #43: (NO WRITE)
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
```

   If any errors are encountered during any phase of the check, error messages are displayed.

See the *Workstation System Manual* for more information about using the HFSCK utility.

Installing Pascal on an HFS Disc

There are three basic steps to installing a bootable Pascal Workstation on an HFS disc.

1. You must install the HFS_DAM (HFS Directory Access Method) module in the initial library (INITLIB). This is accomplished by using the Librarian.
2. Copy the boot files (and other system files) to a specially named directory on the HFS disc. This is done with the Filer.
3. Install the SYSTEM_P file in the "boot area" of the HFS disc by using the OSINSTALL command.

Each step is explained in the sections that follow. Further information about installing a bootable Pascal Workstation on an HFS disc appears in the *Workstation System Manual.*

Adding HFS_DAM to INITLIB

INITLIB contains several modules that provide access to different file systems and file types. As delivered, INITLIB can use LIF discs (LIF_DAM is included) but not HFS discs (HFS_DAM must be added). You will need an initialized disc with enough free space to temporarily store the new INITLIB you are about to create. If you do not have two disc drives you will need to create a memory volume to be used in place of the temporary disc.

1. Insert the ACCESS: disc and invoke the Librarian by typing [L].
2. Insert the BOOT: (or BOOT2:) disc in unit #3: and the "free space" disc in unit #4:.
3. Press [I] and type: `#3:INITLIB`. [Return]
   (You will see a "file not found" message if you forget to include the period.)
4. Press [O] and type: `#4:NEWLIB`. [Return]
5. Transfer all modules **except the last module** from INITLIB to NEWLIB.
   a. Press [E] to Edit. (A new menu will appear.)
   b. Press [U] and type `LAST` [Return].
   c. Press [C] to copy the modules. This will take a moment to complete.
6. When the system finishes copying the modules, remove the BOOT: disc and insert the HFS: disc (or HFS1: disc).
7. Press [A] and type `#3:HFS_DAM`. [Return].
8. Press [=] then [Return] to transfer all of the HFS_DAM modules.
9. When the system finishes copying the module, remove the HFS: disc (or HFS1: disc) and re-insert the BOOT: disc (or BOOT2: disc).
10. Press the Space Bar to continue. The Librarian remembers what you were doing and is now ready to copy the last module.
11. Press [T] to Transfer the LAST module.
12. Press [S] to Stop editing.
13. Press [K] to Keep the output file.
14. Press [Q] to Quit the Librarian.

You have now created a new initialization library that can communicate with HFS discs. You will need to rename this new library (NEWLIB) to the same name as the original INITLIB and use it in place of the original one on the BOOT: disc.

In the next section, where the Pascal files are copied to the HFS disc, be sure to copy the new library you just created and not the INITLIB from the BOOT: disc.

Copying the Pascal Files to an HFS Disc

Once you have initialized your disc with MEDIAINIT and created the hierarchical file system with MKHFS as described earlier in this appendix, you need to make a directory to contain the boot files and subsystems necessary to boot and run the Pascal Workstation from an HFS disc.

As shown earlier, when booting from an HFS disc, your computer expects to find all but the SYSTEM_P file in the directory /WORKSTATIONS/SYSTEM . The SYSTEM_P file is installed by the OSINSTALL utility described later in this appendix.

In the following example, it is assumed that you are installing the Pascal files on an HFS Winchester disc associated with unit #11:.

1. Press [F] to enter the Filer.
2. Press [M] to Make a file or directory.
3. Press [D] to Make a Directory.
4. Type: `#11:/WORKSTATIONS` [Return].
5. Repeat steps 1, 2, and 3 and make this directory: `#11:/WORKSTATIONS/SYSTEM` [Return].

Your disc now has a directory to contain the boot files and your system files.

Now use the Filer to copy your workstation files to the HFS disc. Here we assume that your boot disc (BOOT: or BOOT2:) is in unit #3 and the HFS disc is in unit #11.

1. Press [F] to enter the Filer.
2. Press [F] for Filecopy and type: `#3:SYSTEM_P, #11:/$`
3. Press [F] for Filecopy and type: `#3:INITLIB, #11:/WORKSTATIONS/SYSTEM/$`
4. Press [F] for Filecopy and type: `#3:STARTUP, #11:/WORKSTATIONS/SYSTEM/$`

5. Press [F] for Filecopy and type: `#3:TABLE, #11:/WORKSTATIONS/SYSTEM/$`
6. Press [F] for Filecopy and type: `#3:SWVOL.CODE, #11:/WORKSTATIONS/SYSTEM/$`
7. Press [F] for Filecopy and type: `#3:AUTOSTART, #11:/WORKSTATIONS/SYSTEM/$`

   You now have all of the files necessary to boot Pascal but none of the subsystems such as the Filer, Editor, Compiler, etc.
8. Copy the Pascal subsystems you will want to use to the same directory as the other files on the HFS disc (i.e. `#11:/WORKSTATIONS/SYSTEM/`).

A usable Pascal system is now on the HFS disc but cannot be booted until the SYSTEM_P boot file is installed by the OSINSTALL utility. This is the topic of the next section.

Installing the SYSTEM_P File

In the last section we copied the SYSTEM_P file to the root directory of the HFS disc. The OSINSTALL command will now be used to modify and copy this file to the **boot area** of the disc. Once there, the Pascal Workstation can be booted from an HFS disc.

The OSINSTALL utility is located on the HFS: system disc or the HFS2: system disc (if you ordered your Pascal software on single-sided media). To run the OSINSTALL command:

1. Use the Filer to **Prefix** to the / directory as follows.

   Press [F] [P] `#11:/` [Return].

2. Press [X] and then type: `HFS:OSINSTALL` (or `HFS2:OSINSTALL`) and press [Return].

3. The utility will prompt as follows:

   ```
   OSINSTALL: Check Install Order Remove Quit Zero ?
   ```

4. Type: [I] to Install the SYSTEM_P file. The utility will print a message and prompt:

   ```
   Volume and file to install ?
   ```

5. Type: `#11:/SYSTEM_P` [Return]

6. After a moment the utility will respond:

   ```
   SYSTEM_P installed
   ```

7. Press [Q] to Quit the OSINSTALL utility.

See the *Workstation System Manual* for further information about the Hierarchical File System and the OSINSTALL utility.

Other File System Considerations

If your Pascal Workstation and HP-UX share the same disc, you should read the information in this section. This section explains how HP-UX and the Pascal Workstation handle:

- Context Dependent Files (CDFs) — For information on CDFs, read the chapter "System Management Concepts" found in the *HP-UX System Administrator Manual.*
- Access Control Lists (ACLs) — For information on ACLs, read the chapter "Keeping Your System Secure" found in *Beginner's Guide to HP-UX.*

How HP-UX and the Pascal Workstation Handle CDFs

The differences between the way HP-UX and the Pascal Workstation handle context dependent files are given in the following table:

CDF Task	Pascal Workstation	HP-UX
Listing CDFs	CDFs appear to be directories containing files seen by HP-UX.	CDFs appear to be files according to context.
Creating CDFs	Cannot create CDFs but can deal with them once HP-UX has created them	Can create CDFs
Backing up CDFs	Can store CDFs on a BACKUP archive, but it cannot restore them correctly to an HFS disc. The reason for this is the Pascal Workstation BACKUP program restores CDFs on the archive disc as non-CDF directories containing files.	Can store CDFs on a *cpio* archive, but it cannot restore them correctly to an HFS disc. The reason for this is that *cpio* restores CDFs on the archive disc as non-CDF directories containing files.

How HP-UX and the Pascal Workstation Handle ACLs

The differences between the way HP-UX and the Pascal Workstation handle access control lists are given in the following table:

ACL Task	Pascal Workstation	HP-UX
Listing ACLs	ACLs are not recognized by the Pascal Workstation's HFS driver. The Pascal Workstation only uses the "base" permissions as seen in the Extended listing of the FILER. The only way an access control list can appear on a file owned by the Pascal Workstation is if the HP-UX users "root" or "pws" add them.	ACLs are recognized by HP-UX.
Creating ACLs	Cannot create ACLs. Note that the Pascal Workstation creates files and HFS file systems in a way that will not cause HP-UX's use of ACLs to fail.	Can create ACLs
Backing up ACLs	The Pascal Workstation BACKUP procedure will not save or restore ACL information, although it will save and restore files with ACLs exactly as it does files that do not have them.	Same as the Pascal Workstation.

The Backup Utility

F

There are two different backup utilities for the Pascal Workstation. The TAPEBKUP utility lets you save or restore complete disc images, and is described in the *Workstation System Manual.* The BACKUP utility, described here, is a more generalized utility and can be used to save and restore LIF, WS1.0, SRM, and HFS volumes, directories, and files.

Overview

BACKUP is a utility program that lets you make a backup copy of part or all of your file system. The utility can:

- Create a backup copy of your files on a tape or flexible disc.
- List the contents of a backup.
- Restore all or just parts of the backup to your file system.

The utility is named HFS:BACKUP.CODE (HFS3:BACKUP.CODE if your workstation software was purchased on single-sided media).

General Usage

There are two backup modes. There is a **full backup**, which backs up all files in the specified part of the file system. There is also an **incremental backup**, which backs up only those files that have been created or modified on or after a given date.

It is possible to perform multi-volume backups, single-volume backups, and even back up a single file. The output of BACKUP can even be split over several destination media (if necessary). BACKUP's output can be restored on the Pascal, BASIC, or HP-UX workstations.

To execute the utility, press [x] and type: HFS:BACKUP (or HFS3:BACKUP) [Return]. The prompt will appear on the screen:

```
BACKUP: Full Incr. Restore Table-cont. Quit
```

The five possible commands are:

`Full`	All the files you specify are copied to the destination medium.
`Incr.`	All the files you specify that were created or modified after a given date are copied to the destination medium.
`Restore`	Previously saved files are copied back to your file system.
`Table-cont.`	The contents of a previously created backup are listed.
`Quit`	The utility is exited and control returned to the Main Command Level.

Typing the first letter of each command will invoke the command.

Typically, the backup is written to one or more cartridge tapes. However, flexible discs can be used in place of tapes. If the backup is too big to fit on one medium, you will be prompted to insert another. When more than one medium is used, you will need to mark each one in the order they were used.

Once a backup has been created by the Full or Incremental command, you can examine the contents of the tape with the Table-of-Contents command, and restore some or all of the files with the Restore command.

Creating a Backup

This section describes how to create a backup. It is assumed that a cartridge tape in unit #41: will be used to back up a volume.

Setting the Incremental Date

If you ask for a incremental backup, the following prompt will appear.

```
BACKUP:  Full  Incr.  Restore  Table-cont.  Quit

Incremental backup Date ?
```

The format of this date is the same as is used when setting the date in the **Version** command. The date you type should be the date of the oldest file you wish to have copied to the backup tape. If you do not specify a date, the operation is cancelled.

For example, if you specified `1-Mar-87` as a date, all files created on or after the date `1-Mar-87` will be backed up.

Setting the Destination

The remaining questions are the same for both full and incremental backups. The next question looks like this:

```
BACKUP:  Full  Incr.  Restore  Table-cont.  Quit

Incremental backup Date ? 1-Jan-87

Destination medium for backup ?
```

Here you give a legal volume specification. For example, you could specify `#41:` as the destination. The destination must be a volume specification or unit number of a supported floppy disc or tape device.

Setting the Listing Destination

Results of the backup are listed to the CONSOLE:. Optionally, you can specify the listing to also be sent to the PRINTER: or to a file by typing the backup destination followed by a comma followed by the listing file specification. For example,

```
#41:,PRINTER:
```

or if you wish to save the list on a flexible disc,

```
#41:,#3:LISTFILE
```

Once the destination has been entered, BACKUP will examine the destination media and print a message similar to the following.

```
BACKUP:  Full  Incr.  Restore  Table-cont.  Quit

Incremental backup Date ? 1-Jan-87

Destination medium for backup ?  #41:

Device: HP9144 cartridge tape 700, 1, 0
Logical unit #41 - <no directory>

Checkread the destination (Y/N, default is N) ?
```

The first thing to notice is whether the destination medium is already being used. The second is the option of having the backup operation use the checkread option. Checkread allows you to verify the backup. When verifying, each buffer written to the destination is re-read and compared to the buffer in memory. If there is an error reported during checkread, BACKUP exits at this point **and** the data written on the tape cannot be assumed to be valid.

Changed Your Mind?

Once the checkread option has been answered, you will be asked if everything you have specified is correct. (Note that if BACKUP is being executed by a streaming operation, this question is not asked.)

```
BACKUP:  Full  Incr.  Restore  Table-cont.  Quit

Incremental backup Date ? 1-Jan-87

Destination medium for backup ?  #41:

Device: HP9144 cartridge tape 700, 1, 0
Logical unit #41 - <no directory>

Checkread the destination (Y/N, default is N) ? N

Are you SURE you want to proceed? (Y/N)
```

If you are certain you can use the destination medium, press [y] to begin the backup.

Specifying the Files

Now you are asked for the names of the files to be backed up.

```
Source directory or file to backup ?
```

It is only possible to input one volume, directory, or file specification at a time to this question. However, as long as you type something other than a [Return] to this last question, it will be repeated.

In this example, we will assume your disc is at unit #11:

```
Source directory or file to backup ?  #11: [Return]
```

The backup begins.

Specifying Files for Backup

The format for specifying volumes and filenames is the same as is used when copying files with the Filer. For hierarchical file systems, there are two fundamental ways to specify a file. You can specify an **absolute** path which starts at the root directory (the / directory) or a **relative** path which does not start at the root directory.

Here is a list of the possible directory/file names during backup.

Files from a LIF Volume

Examples: `TST:` , `TST:FILE.TEXT` , `#10:=TEXT` , and `=.CODE`

Files from a SRM or HFS Volume (Relative)

Examples : `TST:` , `#5:TST` , `#5:TST/=TEXT` , `#5:=CODE` , and `TST.TEXT`

Files from a SRM or HFS Volume (Absolute)

Examples : `TST:/`, `#5:/USERS`, `TST:/USERS/=TEXT`, and `/TST`

Files from HP-UX (Relative)

Example : `project/test`

Files from HP-UX (Absolute)

Example : `/users/joe`

Note that you can use the = wildcard, but not the ? or $ wildcards. However, here = has a special meaning when used with hierarchical file systems such as SRM or HFS. If you ask to back up `#5:=` for example, all files and recursively all subdirectories of the current working directory of volume `#5:` are backed up.

The = wildcard character matches directory names and file names recursively, (e.g. "`=.TEXT`" matches all TEXT files in the current directory **and in all subdirectories**. You can also use = in different directory levels in the path name, for instance `=/abc/=`. This will back up each file and directory which has a directory `abc` somewhere in its path. You can mix any number of the three possible types of input in this case. If you input the name of one or more files (e.g `=.TEXT`) those files will be backed up. If the input is the name of a SRM or HFS directory, all files in this directory and recursively all subdirectories will be backed up. If the input is a LIF or WS1.0 volume name, all files on this volume will be backed up.

As the Backup Runs

If the output of a backup doesn't fit onto one tape or floppy disc, you are prompted to insert another one into the same unit.

```
Output medium full, insert medium 2 in #41.
When volume is ready, press <return> to continue.
```

The utility will then check the new medium and prompt you to confirm whether to proceed with this medium or not. With this utility it is possible to have as many tapes/floppies as are required to do a backup. In order to be able to restore all files correctly, you must mark the sequence of the media. This utility does not know anything about the sequence of the backup media and will get "confused" if given media in the wrong order during a restore or table-of-contents operation.

Aborting a Backup

When prompted to insert another medium to finish the backup, you should always do so, otherwise data may be lost. This is due to the internal buffer mechanism BACKUP uses. Even though a file may be listed as being backed up, it may not yet be written on the output medium itself.

It is important not to [Stop] the backup procedure at any time, otherwise backup data is lost. Only use [Stop] if something went wrong. [Reset] is disabled during the execution of this utility.

Table Of Contents

Table-of-contents lists all the file names on a backup medium. When you select the command, the questions look something like this:

```
BACKUP:  Full  Incr.  Restore  Table-cont.  Quit

List content of what medium ?
```

Here you specify the unit which contains a backup. It is also possible to change the listing output to something other than CONSOLE:, just the same as for backup and restore. For example,

`#41:,PRINTER:` or `#3:,LISTFILE`

After specifying the volume, you can specify a subset of files to be listed.

```
BACKUP:  Full  Incr.  Restore  Table-cont.  Quit

List content of what medium ? #41:

List what files (default is all) ?
```

You can press Return to list all of the files that are on the tape or specify (with a wildcard) some subset of files to list.

If several media were used when the back up was created, you will need to insert each medium in its correct order. A message like this will appear:

```
BACKUP:  Full  Incr.  Restore  Table-cont.  Quit

List content of what medium ? #41:
List what files (default is all) ?

At end of input medium number 1, insert next medium in #41.
When new volume is ready, press <return> to continue.
```

Remove the current medium and insert the next. If using a cartridge tape, be sure to wait a minute or two to let the tape drive load the tape first.

Restoring a Backup

When you select Restore from the utility's menu, you will be asked:

```
BACKUP:  Full  Incr.  Restore  Table-cont.  Quit

Source medium for restore ?
```

Here you type the volume specification for the tape or disc containing the backup (the source of the restore).

Unconditional Restore of Files

Normally only files which do **not** exist on the destination file system (or do exist but are older than the one on the backup) are restored.

```
BACKUP:  Full  Incr.  Restore  Table-cont.  Quit

Source medium for restore ? #41:
Restore unconditionally (Y/N, default N) ?
```

If you really want to overwrite a newer version of a file with the file from a former backup, you can unconditionally restore the file. But note that this flag cannot be set for each file individually, it pertains to all restored files.

Listing the Restored Files

When restoring, a listing of restored files is output. It is possible to change where the listing will go. This is done the same way it is done when creating the backup, but here it gets specified together with the source medium. For example, `#41:,PRINTER:`.

Renaming During Restoring

You can restore all files from one volume to another one. This is enabled when the following question is answered with [Y].

```
Rename destination volume (Y/N, default Y) ?
```

In this case every time a new volume name is found on the restore media you will be asked for the name of the new destination volume.

Resynchronizing

When restoring, if a file cannot be read from the backup media, the restore operation will fail. If you suspect there may be problems in restoring the backup, you can tell the utility to resynchronize (i.e. skip over any unreadable file to the next possible file).

```
Resynchronize (Y/N, default N) ?
```

This option is discussed later in this appendix.

Specifying Files to be Restored

You can now specify which files you want to restore.

```
Source files to restore (default is all) ?
Source files to restore ?
```

Here too, wildcards can be used to specify the files to be restored, for example `=.TEXT` will restore all files ending with ".TEXT".

The next prompt asks:

```
Are you SURE you want to proceed? (Y/N)
```

After answering with [Y], the restore is started. If you chose the rename mode, you are asked:

```
Destination volume/directory to restore to (default is volume name)?
```

If you do not type in a name, the name found on the tape is used. If this volume or directory does not exist, you will be asked for a new specification until you type an existing volume name.

If you chose the non-renaming mode, be aware that the required volumes must exist. If a volume or directory does not exist those files are **not** restored in this mode, but the restore operation will continue.

Whether renaming or not, a list of all files which are restored as well as a list of all files not restored is output. In the case where files from an HP-UX backup are being restored, you will be asked for a new volume name until you specify a valid one.

Restoring and Renaming

You can restore a backup with or without renaming. An automated restore (non-renaming restore) is only possible if **nothing** has changed in the configuration since the last backup. It is also mandatory that the current working directories on all volumes are the same as during backup if the backup was "relative" or if volume names were used.

Renaming

During restore on the Pascal Workstation you can perform a special renaming procedure. For every new volume to be restored you will be asked for the destination volume/directory.

From LIF to LIF

All files are restored onto the destination LIF volume.

From LIF to SRM or HFS (Relative)

All files are restored into the given directory. If only a volume name is given, the files will be restored onto that volume into the "current working" directory.

From LIF to SRM or HFS (Absolute)

All files are restored into the given directory.

From SRM or HFS or HP-UX (Relative) to LIF

The path is removed from the file name. You must be aware of naming conflicts. If there is a naming conflict the result depends on the options you chose. For example, if you have restored all files unconditionally the last file from the tape will be in the LIF directory. If not, then the first file from the tape with a given name will be on the LIF directory. This is due to the fact that in unconditional mode a file only gets restored if it does not already exist in the destination directory or if the one in the destination directory is older than the one on the tape. A file that is restored to LIF gets the current system date as the creation date, so it (normally) will always be newer than other files on the tape.

From SRM or HFS or HP-UX (Relative) to SRM or HFS (Relative)

All files are restored into the given directory. If only a volume name is given the files are restored onto that volume into the "current working" directory. The path on the file name will be kept. For example:

Original File Name:	`TST:tst1/tst2/TST.TEXT`
Directory to Restore to:	`ABC:xyz`
Resulting File Name:	`ABC:xyz/tst1/tst2/TST.TEXT`

In this case `ABC:xyz` must be an existing directory. The directories `xyz/tst1` and `xyz/tst1/tst2` are created if they don't exist already.

From SRM or HFS or HP-UX (Relative) to SRM or HFS (Absolute)

All files are restored into the given directory. The path on the file name will be kept.

From SRM or HFS or HP-UX (Absolute)

If something was backed up absolute it will always be restored absolute no matter how the destination volume/directory is given. When the destination is a LIF volume the path is stripped off. For example:

Original File Name:	`TST:/tst1/tst2/TST.TEXT`
Directory to Restore to:	`ABC:/xyz`
Resulting File Name:	`ABC:/tst1/tst2/TST.TEXT`
	The directories `/tst1` and `/tst1/tst2` are created if they do not already exist.

If you restore a backup made on the Pascal Workstation to HP-UX, you have two possibilities, either use renaming mode or not. In the renaming mode you have to input a new name for every single file on the tape. In the other case, for each volume entry a directory "<*volume name*>:" will be created in the current working directory and all files will be restored relative or absolute depending on the mode used during backup.

Resynchronizing

This utility has a resynchronizing capability which is automatically set during table of contents and can be set for restore (discussed later). This is very useful when, for example, a tape/floppy from a multi-media backup is bad. In this situation you can directly start the restore with a medium other than the first medium of a sequence. Obviously, you will lose data from at least one file.

If the resynchronize mode is chosen and tapes are inserted in the wrong order you may get undesired results. The last file from a medium may contain wrong data when restored. It is also possible in this case that some files get overwritten. In other words, you have to be very careful when in this mode.

If you insert the tapes/floppies in the correct order and there is no problem with one of the tapes/floppies, resynchronize has no effect. If something is wrong and BACKUP tries to resynchronize you will see the following message:

```
Out of phase; resynchronizing.
```

And after resynchronize was successful (a valid file header could be found):

```
Resynchronized after skipping xxxxx bytes.
```

At the end of restore you will see

```
Unable to restore at least xxx files.
```

If something goes wrong, for example you use the wrong media sequence or use a backup from HP-UX that was created by a `cpio` without a `-c` option, you will see the message:

```
Out of phase--get help.
```

This means the utility can no longer determine what to do. Consider making another backup. Be sure to mark each tape or disc in the order it was used.

Special Considerations

Here are some special considerations to watch for when using BACKUP.

Tips for SRM Backups

Passwords

Backups made from an SRM do not include any information about passwords associated with files or directories. When these files or directories are restored, any password protection must be performed manually.

To back up a file from SRM you must specify the password (if it has one) together with the file name. If there are passwords in different directory levels, however, it gets quite difficult to do recursive backups. You then must do a backup on a file by file or subdirectory by subdirectory basis in this case. If you have the SRM volume password you can back up all files. See your SRM Administrator for the volume password.

Duplicate Links

SRM files which are linked to other files can not be restored as linked files. However, HFS files which are linked will have links restored if the files are restored to an HFS volume.

Dates

Files restored to the SRM will be given the date of the restore operation and not the original date of the file.

Tips for HFS Backups

With this utility all files can be backed up from an HFS disc, even those that were not created by a Pascal or BASIC user. This is equivalent of doing a backup on HP-UX as the super-user. During restore however, there are some special checks. Only if a restore is done to an HFS disc can files be restored regardless of permissions. Files that are not owned by Pascal or BASIC, or don't have the appropriate read permission can't be restored to a non-HFS disc.

Backups created by the Pascal workstation that are restored to an HFS disc get the **uid**, **gid**, **mode** of the workstation and last **modification date** of the file. This is similar to the use of `cpio -m` on an HP-UX workstation.

HP-UX "special files" (such as found in the `/dev` directory) will be backed up, but cannot be restored using this utility. They can be restored by HP-UX.

Tape Verification

There is no verification equivalent to the TAPEBKUP's **Verify** command. Instead there is a checkread verify. In this mode, each buffer written to the destination is checkread and compared with the original data. The backup is much slower in this mode. By default, checkread is **not** performed. If you want to verify the output of a backup, use the TAPEBKUP verify mode. If you **really** want to verify, use the checkread option.

Output of BACKUP

The output medium of BACKUP is a cartridge tape or flexible disc (removable media). Supported by this utility are all CS/80 and Subset/80 devices. The destination for backup and source for restore (and table of contents) can be any of the floppy discs and tape drives supported by the Pascal Workstation.

The output format of the BACKUP utility is the same as is used by HP-UX (the `cpio` format using the `-c` option). When the medium is a tape cartridge, the output is blocked correctly for the HP-UX `tcio` command.

Volume Information

To distinguish files from different Pascal Workstation volumes additional information is written on the output media. For each volume backed up, a directory entry will exist on the tape. This entry will also contain information about the BASIC MSUS of the volume. These entries will be used by the BASIC Workstation when restoring, but will be ignored by the HP-UX and the Pascal workstations.

On the Pascal Workstation this volume info serves to distinguish files from different volumes. During an automatic restore this volume name will be the name of the destination volume. When doing a restore in renaming mode this is the default destination volume.

On an HP-UX workstation, if `cpio` is called with the `-d` option, a directory will be created with the name of volume found on the backup. The name of the volume is not stored in the file name on the backup. This can result in file name collisions when restoring under HP-UX. On the Pascal Workstation all file names are unique, because the Pascal Workstation knows of the special meaning of the volume information in front of the files of a specific volume.

The volume information is always kept on the BACKUP output, no matter whether there is a single or multiple volume backup. This is not only for consistency reasons, but also to make it easier to distinguish a backup made on HP-UX from one made on the Pascal Workstation.

Directories, except for LIF and WS1.0 volumes, are also backed up. For HFS and SRM this will result an empty directory entry on the tape. This is different from cpio on HP-UX; there the directory contents itself is also backed up. During restore however the contents of the directory entry is totally ignored on HP-UX and the Pascal Workstation, therefore it is not output by this utility. New SRM or HFS directories will be created during an automatic restore, even if they are empty. On the Pascal Workstation directories are always created during restore.

The Pascal Workstation backup will assume use of ASCII format headers (`cpio -c` option). HP-UX `cpio` backups must be done with this option if they are to be restored to a Pascal Workstation.

A backup on a cartridge tape made on the Pascal Workstation can only be restored on HP-UX using `tcio` with `cpio` (for example: `tcio -i /dev/rct | cpio -icvdmlx`).

Index

a

b

C

d

e

f

g

h

i

k

l

m

n

o

p

q

r

S

t

u

V

W

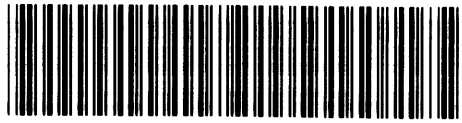
x

y

z

Notes:

HP Part Number
98615-90042
Microfiche No. 98615-99042
Printed in U.S.A. E0389



98615-90636
For Internal Use Only